CURSO PRÁTICO 68 DE PROGRAMAÇÃO DE COMPUTADORES

INPUT

PROGRAMAÇÃO BASIC - PROGRAMAÇÃO DE JOGOS - CÓDIGO DE MÁQUINA

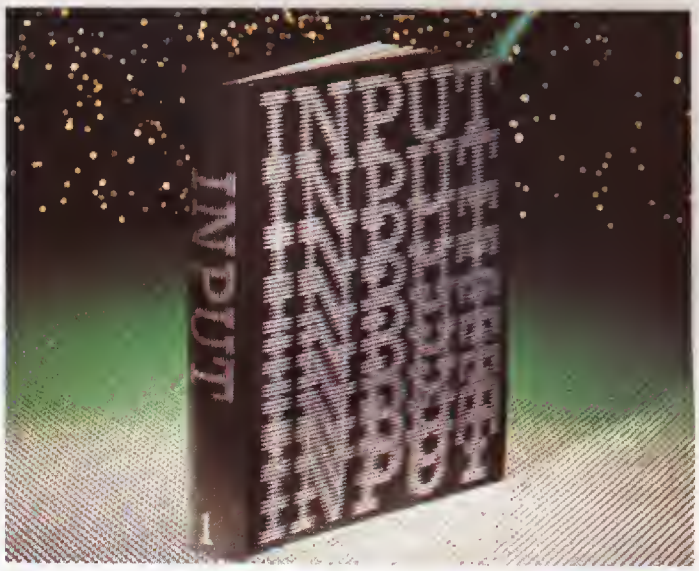
NOVA CULTURAL

Cz$ 50,00

INPUT

Vol. 5 N.º 68

NESTE NÚMERO



PLANO DA OBRA

"INPUT" é uma obra editada em fascículos semanais, e cada conjunto de 15 fascículos compõe um volume. A capa para encadernação de cada volume estará à venda oportunamente.

COMPLETE SUA COLEÇÃO

Exemplares atrasados, até seis meses após o encerramento da coleção, poderão ser comprados, a preços atualizados, da seguinte forma: 1. PESSOALMENTE — Por meio de seu jornaleiro ou dirigindo-se ao distribuidor local, cujo endereço poderá ser facilmente conseguido junto a qualquer jornaleiro de sua cidade. Em **São Paulo**, os endereços são: rua Brigadeiro Tobias, 773, Centro; avenida Industrial, 117, Santo André; e no **Rio de Janeiro**: avenida Mem de Sá, 191/193, Centro. 2. POR CARTA — Poderão ser solicitados exemplares atrasados também por carta, que deve ser enviada para DINAP — Distribuidora Nacional de Publicações — Números Atrasados — Estrada Velha de Osasco, 132, Jardim Teresa — CEP 06000 — Osasco — SP. Não envie pagamento antecipado. O atendimento será feito pelo reembolso postal e o pagamento, incluindo as despesas postais, deverá ser efetuado ao se retirar a encomenda na agência do Correio. 3. POR TELEX — Utilize o nº (011) 33 670 DNAP.

Em **Portugal**, os pedidos devem ser feitos à Distribuidora Jardim de Publicações, Lda. — Qta. Pau Varais, Azinhaga de Fetais — 2 685, Camarate — Lisboa; Apartado 57 — Telex 43 069 JARLIS P.

Atenção: Após seis meses do encerramento da coleção, os pedidos serão atendidos dependendo da disponibilidade do estoque.

Obs.: Quando pedir livros, mencione sempre título e/ou autor da obra, além do número da edição.

COLABORE CONOSCO

Encaminhe seus comentários, críticas, sugestões ou reclamações ao Serviço de Atendimento ao Leitor — Caixa Postal 9442, São Paulo — SP.

EDITOR
RICHARD CIVITA

NOVA CULTURAL

Presidente
Flávio Barros Pinto
Diretoria
Carmo Chagas, Iara Rodrigues,
Pierluigi Bracco, Plácido Nicoleto,
Roberto Silveira, Shozi Ikeda,
Sônia Carvalho

REDAÇÃO
Diretor Editorial: Carmo Chagas
Editores Executivos:
Antonio José Filho, Berta Sztark Amar
Editor Chefe: Paulo de Almeida
Editoras Assistentes: Ana Lúcia B. de Lucena,
Marisa Soares de Andrade
Chefe de Arte: Carlos Luiz Batista

Assistentes de Arte: Dagmar Bastos Sampaio,
Grace Alonso Arruda, Monica Lenardon Corradi

Colaboradores
Consultor Editorial Responsável:
Dr. Renato M. E. Sabbatini
(Diretor do Núcleo de Informática Biomédica da Universidade Estadual de Campinas)
Execução Editorial: DATAQUEST Assessoria em Informática Ltda., Campinas, SP
Tradução, adaptação, programação e redação:
Abílio Pedro Neto, Aluísio J. Dornellas de Barros,
Marcelo R. Pires Therezo, Marcos Huascar Velasco,
Raul Neder Porrelli, Ricardo J. P. de Aquino Pereira
Coordenação Geral: Rejane Felizatti Sabbatini

COMERCIAL
Diretor Comercial: Roberto Silveira
Gerente Comercial: Flávio Maculan
Gerente de Circulação: Denise Mozol
Gerente de Propaganda e Publicidade: José Carlos Madio
Gerente de Pesquisa e Análise de Mercado:
Wagner M. P. Nabuco de Araújo

CLC

A Editora Nova Cultural Ltda. é uma empresa do Grupo CLC — Comunicações, Lazer e Cultura
Presidente: Richard Civita
Diretoria: Flávio Barros Pinto, João Gomez,
Menahen M. Politi, Renê C. X. Santos,
Stélio Alves Campos

Edição organizada pela Editora Nova Cultural Ltda.
Av. Brigadeiro Faria Lima, nº 2000 - 3º andar
CEP 01452 - São Paulo - SP - Brasil
(Artigo 15 da Lei 5 988, de 14/12/1973).
Esta obra foi composta na AM Produções Gráficas Ltda.
e impressa na Divisão Gráfica da Editora Abril S.A.

LOGO: ALÉM DA TARTARUGA

- EXPRESSÕES MATEMÁTICAS
- VARIÁVEIS E NÚMEROS
- NOMES E SENTENÇAS
- O QUE SÃO LISTAS
- COMANDOS PARA LISTAS

Os recursos do LOGO não se esgotam em seus eficientes comandos gráficos. Manipular palavras, representar dados, efetuar cálculos matemáticos — tudo isso o LOGO pode, e muito mais.

Neste terceiro e último artigo da série sobre programação LOGO, examinaremos as características fundamentais dessa linguagem. Na realidade, seguimos o caminho inverso do que é usual no aprendizado de outras linguagens, já que abordamos em primeiro lugar os comandos gráficos. Mas esta é a maneira mais fácil de entrar em contato com o universo da programação — principalmente para crianças. Aliás, justamente por causa da popularidade de seus recursos gráficos, muita gente pensa que o LOGO é apenas uma "linguagem gráfica", própria para crianças.

Nada mais longe da verdade. Como o LISP, seu antecessor, o LOGO foi projetado de modo a simplificar a manipulação de palavras e listas de palavras e frases. Essa característica não impede, contudo, que complexos programas de Inteligência Artificial sejam desenvolvidos com a ajuda do LOGO.

Ao estudar linguagens imperativas, inclusive o BASIC, começamos por analisar seus elementos básicos, tais como as formas de representação de dados

O que é um micromundo LOGO?

Um micromundo é um conjunto conexo de procedimentos, escritos em LOGO, que estruturam uma determinada situação ou ambiente de resolução de problemas cognitivos.

Esses procedimentos podem ser combinados em outros procedimentos, de forma cada vez mais complexa, imitando, assim, o desenvolvimento natural do *conhecimento* da criança sobre uma parte do mundo.

Exemplificando: podemos construir um micromundo em que a tela é a gaiola de um coelhinho, com lugares para ele brincar, dormir, comer etc. Usando o LOGO, a criança e o professor elaboram vários procedimentos para desenhar o ambiente e seus objetos, fazer o coelhinho se movimentar, levá-lo até a área de dormir etc., de maneira progressiva e hierárquica. Esse micromundo pode ser tão complexo e completo quanto se queira.

(constantes e variáveis), os comandos de impressão etc. É o que faremos agora com a linguagem LOGO.

Como nos artigos anteriores, apresentaremos sempre duas versões dos programas: a primeira, identificada pelo logotipo Apple, corresponde ao padrão MIT LOGO, em inglês, adotado universalmente com poucas variações; a segunda, identificada pelo logotipo do MSX, corresponde à versão nacional, chamada BRASLOGO, desenvolvida pela Unicamp para a Itautec e adotada por outros fabricantes. Ao final do artigo, damos a tradução dos comandos usados para o MLOGO, versão nacional muito difundida, destinada ao Apple.

MATEMÁTICA EM LOGO

Como qualquer linguagem de programação que se preze, o LOGO dispõe de amplos recursos destinados a cálculos matemáticos com diferentes níveis de complexidade; não se restringe às quatro operações aritméticas.

O LOGO não faz distinção entre números inteiros e reais (números que podem ser fracionários, como 1.2, 3.14156 etc.). Em alguns interpretadores, entretanto, a falta do ponto decimal nos argumentos de uma operação força um resultado inteiro.

Os operadores matemáticos são iguais aos da linguagem BASIC: soma (+), subtração (−), divisão (/), multiplicação (*), parênteses etc. A ordem de execução das operações também é a mesma. Usando, por exemplo, o comando **PRINT** (**ESCREVA**, na versão em português), com função idêntica à que tem no BASIC, podemos fazer:

```
PRINT 2 + 2*3
```

```
ESCREVA 2 + 2*3
```

O resultado da operação será igual a 8, pois, nesta expressão, efetua-se a multiplicação em primeiro lugar.

As expressões matemáticas podem ser colocadas em outros comandos do LOGO, como no desenho de gráficos:

```
FORWARD (10 + 120)/3.14
```

```
PARAFRENTE (10 + 120)/3.14
```

Os procedimentos também comportam expressões matemáticas:

```
TO PI2
PRINT 3.14156*2
END
```

```
APRENDA PI2
ESCREVA 3.14156*2
FIM
```

Feito isso, sempre que digitarmos o comando **PI2** pelo teclado, obteremos a seguinte resposta:

```
2.28312
```

Uma expressão pode utilizar variáveis. Como você deve estar lembrado, as variáveis recebem nomes, que podem ser de qualquer comprimento, e são identificadas por dois pontos. Assim, um procedimento destinado a mostrar o cubo de um número qualquer seria escrito da seguinte maneira:

```
TO CUBO :NUMERO
PRINT :NUMERO*:NUMERO*:NUMERO
END
```

```
APRENDA CUBO :NUMERO
ESCREVA :NUMERO*:NUMERO*:NUMERO
FIM
```

O procedimento criado exige um argumento numérico de entrada, e responde imprimindo na tela o cubo do mesmo:

```
CUBO 3
27
CUBO 10
1000
```

Eis aqui outro procedimento, que calcula a média de dois números:

```
TO MEDIA :N1 :N2
PRINT (:N1 + :N2)/2
END
```

```
APRENDA MEDIA :N1 :N2
ESCREVA (:N1 + :N2)/2
FIM
```

Esse procedimento requer a presença de dois argumentos:

```
MEDIA 2 4
3
MEDIA 1 2
1.5
```

Em alguns interpretadores LOGO, a resposta ao segundo comando poderá ser igual a 1, ou seja, o resultado será mostrado em forma inteira, pois os argumentos de entrada estavam expressos desse modo. Se isso ocorrer, podemos forçar uma operação com resultados fracionários, escrevendo:

```
MEDIA 1.0 2.0
```

Não se esqueça de colocar os dois argumentos para o procedimento **MEDIA**. Caso contrário, o interpretador reclamará, exibindo uma mensagem de erro.

Observe que, em todas as expressões aritméticas examinadas, o símbolo da operação matemática a ser realizada aparece entre os argumentos da operação (por exemplo, 2+2). Essa notação, chamada *infixa*, é a mais usada em linguagens imperativas derivadas do FORTRAN. Mas existem dois outros tipos de notação: a *sufixa*, ou notação polaca inversa (RPN), que é usada em linguagens como o FORTH (a mesma operação acima seria escrita 2 2 +), e a *prefixa*, que também é aceita pela linguagem LOGO. Nesse tipo de notação, o comando precede os argumentos que usa.

Para realizar operações prefixas no LOGO, empregam-se comandos já existentes no interpretador. Por exemplo:

```
SUM 6 7
DIVIDE 3 1.2
PRODUCT 3 4
```

Em alguns interpretadores LOGO, esses comandos recebem nomes diferentes, como **QUOTIENT** em vez de **DIVIDE**, ou **MULTIPLY** em vez de **PRODUCT**. Pode acontecer, também, que o interpretador não contenha o procedimento **SUBTRACT**. Nesse caso, basta usar **SUM**, com um número negativo.

```
SOMA 6 7
QUOCIENTE 3 1.2
PRODUTO 3 4
```

Se um desses comandos não existir no interpretador LOGO que está usando, deverá fazê-lo **APRENDER** o novo procedimento.

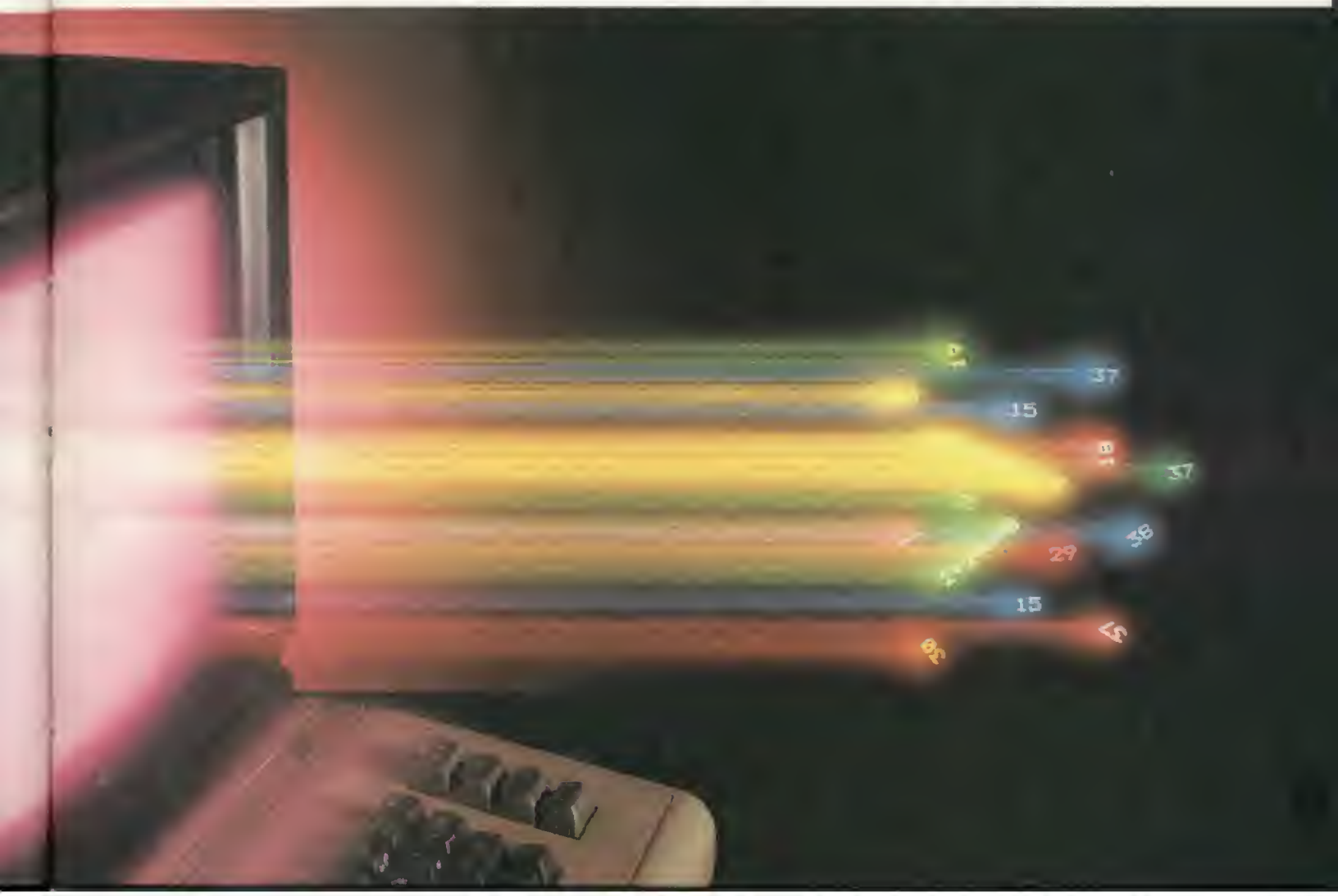
Note que é possível utilizar funções dentro de funções, como em:

```
DIVIDE SUM 3 2 SUM 4 5
```

```
QUOCIENTE SOMA 3 2 SOMA 4 5
```

que corresponde à notação infixa:

```
PRINT (3+2)/(4+5)
```



Não tente fazer o mesmo, porém, com os procedimentos **CUBO** e **MEDIA**, desenvolvidos anteriormente, pois eles apenas imprimem os respectivos resultados, não tendo meios de "passá-los" a uma outra função. Experimente, por exemplo, digitar a seguinte expressão para ver o que acontece:

```
CUBO MEDIA 3.4 7.88
```

Para retornar um resultado, o procedimento precisa incluí-lo na lista de variáveis na chamada. Uma alternativa mais simples consiste em usar o comando **OUTPUT** (ou **ENVIE**, na versão para a língua portuguesa):

```
TO MEDIA :N1 :N2
OUTPUT (:N1 + :N2)/2
END
```

```
APRENDA MEDIA :N1 :N2
ENVIE (:N1 + :N2)/2
FIM
```

Agora, você poderá recorrer ao procedimento **MEDIA** como argumento de outro procedimento.

FUNÇÕES MATEMATICAS

O LOGO possui várias funções matemáticas que facilitam o cálculo de expressões mais complexas — algébricas, trigonométricas etc.

REMAINDER	Resto de uma divisão
ROUND	Arredondamento
ABS	Absoluto
INT	Inteiro
SQRT	Raiz quadrada
SIN	Seno
COS	Co-seno

RESTO	Resto de uma divisão
INTEIRO	Inteiro
RAIZQ	Raiz quadrada
SENO	Seno
COS	Co-seno

Só alguns interpretadores têm o conjunto completo dessas funções.

O LOGO conta ainda com um gerador de números aleatórios, útil em uma série de aplicações, inclusive jogos.

Eis um programa que simula dez lançamentos de um dado:

```
TO DADOS
REPEAT 6 [PRINT RANDOM 6]
END
```

```
TO DADOS
REPITA 6 [ESCREVA SORTEIEATÉ 6]
FIM
```

Para inicializar uma seqüência aleatória, utiliza-se o comando **RANDOMIZE** (**REPRODUZA**, na versão BRASLOGO).

PALAVRAS

É possível especificar palavras (constantes alfanuméricas) em LOGO, identificando-as com o serial de aspas. Se quiser, por exemplo, imprimir no vídeo a palavra COMPUTADOR, digite:

```
PRINT "COMPUTADOR
```

```
ESCREVA "COMPUTADOR
```

Note que não se fecham as aspas, como no BASIC. Se você se esquecer de colocar as aspas antes da palavra, a máquina entenderá que COMPUTADOR é o nome de um número ou de procedimento criado anteriormente; não o encontrando, exibirá uma mensagem de erro.

Como o comando de impressão aceita apenas um argumento, as formas que se seguem também são incorretas:

```
PRINT "BOM DIA
PRINT "BOM "DIA
```

```
ESCREVA "BOM DIA
ESCREVA "BOM "DIA
```

No primeiro caso, o interpretador LOGO informará que ainda não conhece DIA. No segundo, dirá que não sabe o que fazer com "DIA (ou, dependendo do computador, que há um argumento a mais). É importante lembrar que um espaço em branco sempre indica, em linguagem LOGO, o fim de uma palavra, e que não podemos colocar mais de uma palavra após um sinal de aspas.

Para imprimir duas palavras usando um mesmo comando, precisamos antes juntá-las em um único argumento. Existe um comando primitivo que faz isso:

```
PRINT WORD "BOM "DIA
```

```
ESCREVA PALAVRA "BOM "DIA
```

O comando **WORD** (ou **PALAVRA**) toma dois argumentos — que são letras ou palavras precedidas de aspas — e os reúne em uma só palavra. Esta é passada para o comando **PRINT** (ou **ESCREVA**), que então a coloca na tela.

O LOGO possui diversos comandos extremamente poderosos para lidar com palavras. Para extrair letras de uma palavra, utilizam-se estes comandos:

MICRO DICAS

TRADUÇÃO PARA O MLOGO

Os comandos do MLOGO, para o Apple, correspondentes aos comandos do BRASLOGO deste artigo, são:

BRASLOGO	MLOGO
ESCREVA	MOSTRAR
ATRIBUA	FAÇA
COLOQUE	—
SOMA	SOME
PRODUTO	PRODUTO
QUOCIENTE	QUOC
RESTO	RESTO
RAIZQ	RQD
INTEIRO	INT
—	APROX
REPRODUZA	RESORTEIE
SORTEIEATÉ	SORTEIE
SEN	SEN
COS	COS
ARCTAN	ATAN
PALAVRA	PALAVRA
SENTENÇA	SENTENÇA
PRIMEIRO	PRIMEIRO
SEMPRIMEIRO	SEMPRIMEIRO
ÚLTIMO	ULTIMO
SEMÚLTIMO	SEMULTIMO
ENVIE	SAIDA

FIRST - Extrai a primeira letra.
LAST - Extrai a última letra.
BUTFIRST - Extrai todas as letras, menos a primeira.
BUTLAST - Extrai todas as letras, menos a última.

Como exemplo, tente os comandos:

```
PRINT FIRST "ABCD
PRINT LAST "ABCD
PRINT BUTFIRST "ABCD
PRINT BUTLAST "ABCD
PRINT FIRST BUTFIRST "ABCD
PRINT LAST BUTLAST "ABCD
PRINT WORD FIRST "ABCD LAST
   "ABCD
```

Note como os comandos são encadeados: **FIRST BUTFIRST** extrai a segunda letra da palavra ABCD; **LAST BUTLAST** extrai a penúltima palavra etc.

O pequeno procedimento recursivo que mostramos a seguir é capaz de extrair todas as letras de uma palavra:

```
TO LETRAS :PALAVRA
IF :PALAVRA = " THEN STOP
PRINT FIRST :PALAVRA
LETRAS BUTFIRST :PALAVRA
END
```

O procedimento ilustra o conceito de *palavra vazia* — ou seja, palavra sem nenhum caractere. A chamada recursiva de **LETRAS**, na quarta linha, usa como argumento o **BUTFIRST** da palavra que foi entrada. Esta vai sendo reduzida até não ter mais nenhuma letra. A recursão só se encerra quando o teste executado pelo **IF** da segunda linha é verdadeiro. A palavra vazia é assinalada pelas aspas e um espaço em branco.

PRIMEIRO - Extrai a primeira letra.
ÚLTIMO - Extrai a última letra.
SEMPRIMEIRO - Extrai todas as letras, menos a primeira.
SEMÚLTIMO - Extrai todas as letras, menos a última.

Como exemplo, tente os comandos:

```
PRINT PRIMEIRO "ABCD
PRINT ULTIMO "ABCD
PRINT SEMPRIMEIRO "ABCD
PRINT SEMULTIMO "ABCD
```

```
PRINT PRIMEIRO SEMPRIMEIRO
   "ABCD
PRINT ULTIMO SEMULTIMO "ABCD
PRINT PALAVRA PRIMEIRO "ABCD
ULTIMO "ABCD
```

Note como os comandos são encadeados: **PRIMEIRO SEMPRIMEIRO** extrai a segunda letra da palavra ABCD; **ÚLTIMO SEMÚLTIMO** extrai a penúltima palavra e assim por diante.

O pequeno procedimento recursivo mostrado a seguir é capaz de extrair todas as letras de uma palavra:

```
APRENDA LETRAS :PALAVRA
SE :PALAVRA = " ENTAO [PARE]
ESCREVA PRIMEIRO :PALAVRA
LETRAS SEMPRIMEIRO :PALAVRA
FIM
```

O procedimento ilustra o conceito de *palavra vazia* — ou seja, palavra sem nenhum caractere. A chamada recursiva de **LETRAS**, na quarta linha, usa como argumento o **SEMPRIMEIRO** da palavra que foi entrada. Esta vai sendo reduzida até não ter mais nenhuma letra. A recursão só se encerra quando o teste executado pelo **SE** da segunda linha é verdadeiro. A palavra vazia é assinalada pelo sinal de aspas seguido de um espaço em branco.

ATRIBUIÇÃO

O comando de atribuição — **LET**, em linguagem BASIC — também existe em LOGO. Com ele, podemos armazenar números ou resultados de expressões em variáveis, e dar-lhes nome.

```
MAKE "IDADE 15
```

```
ATRIBUA "IDADE 15
```

Podemos ainda colocar uma palavra em variável:

```
MAKE "NOME "JOAQUIM
```

```
ATRIBUA "NOME "JOAQUIM
```

Experimente verificar o conteúdo das variáveis **NOME** e **IDADE**, com:

```
PRINT :NOME
PRINT :IDADE
```

```
ESCREVA :NOME
ESCREVA :IDADE
```

Como você provavelmente deve ter observado, usamos dois pontos, e não aspas, quando estamos nos referindo a uma variável já criada. Esta é uma característica fundamental do LOGO. Nessa linguagem, ao contrário do BASIC, é necessário diferenciar *explicitamente* as referências à variável (**"NOME**) e ao seu conteúdo (**:NOME**).

Na realidade, todos os elementos do LOGO são definidos do mesmo modo, e armazenados na mesma estrutura. Tanto os comandos (programas) quanto os dados são *palavras*. Números também são tratados como palavras. Assim, é perfeitamente possível fazer-se algo como:

```
PRINT "2.34 + "19.87
PRINT BUTFIRST 12.345
```

```
ESCREVA "2.34 + "19.87
ESCREVA SEMPRIMEIRO 12.345
```

Eis aqui um pequeno programa que ilustra essa característica do LOGO. Ele imprime uma tabela contendo o quadrado e a raiz quadrada de todos os números inteiros, de **N1** a **N2**:

```
TO TABELA :N1 :N2
IF :N1=:N2 THEN STOP
PRINT WORD :N1 :N1*N1 SQRT(:N1)
TABELA :N1+1 :N2
END
```

```
TO TABELA :N1 :N2
SE :N1=:N2 ENTAO [PARE]
ESCREVA PALAVRA :N1 :N1*N1
RAIZQ(:N1)
TABELA :N1+1 :N2
FIM
```

Para fazer o procedimento **TABELA** chamar a si mesmo, usamos a recursão incrementando a variável de início **:N1**. A segunda linha do procedimento verifica o valor dessa variável, parando se ela já tiver atingido o máximo desejado (**:N2**). Caso contrário, imprime na tela uma linha contendo o número, seu quadrado e sua raiz quadrada. Como o comando de escrita aceita um só argumento, precisamos usar um "aglutinador" dos diferentes números, colocando-os em uma única lista, por meio do comando **WORD** (ou **PALAVRA**).

LISTAS

Todo o potencial do LOGO se revela quando examinamos a última de suas estruturas básicas de dados: a *lista*, um conjunto de palavras indicado por meio de colchetes. Por exemplo:

```
[TIGRE LEAO GATO]
```

é uma lista com três elementos, separados por brancos.

Podemos dar nome às listas:

```
MAKE "FELINOS [TIGRE LEAO GATO]
MAKE "IMPARES [1 3 5 7 9]
MAKE "COLEGAS [JOAQUIM 15 MARIO
14]
```

```
ATRIBUA "FELINOS [TIGRE LEAO
GATO]
ATRIBUA "IMPARES [1 3 5 7 9]
ATRIBUA "COLEGAS [JOAQUIM 15
MARIO 14]
```

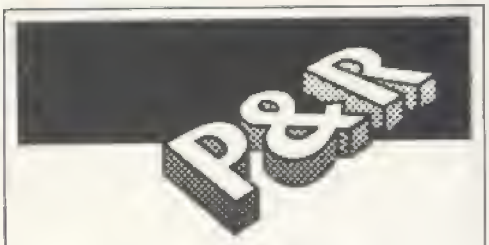
Se usarmos o comando **PRINT** (ou **ESCREVA**) seguido do nome de uma lista, obteremos seu conteúdo completo. Uma lista pode conter outras listas:

```
MAKE "CARNIVOROS [FELINOS LOBO
RAPOSA]
```

```
ATRIBUA "CARNIVOROS [FELINOS
LOBO RAPOSA]
```

Todas as funções utilizadas para manipulação de palavras aplicam-se também a listas. Por exemplo:

```
PRINT FIRST FELINOS
```



Quais as etapas para o desenvolvimento de uma nova linguagem?

Desenvolver uma nova linguagem não é difícil — tanto que existem centenas delas que nunca passaram do estágio de mera "brincadeira" científica para uma efetiva implementação e distribuição no mercado.

Em primeiro lugar, deve-se definir se a linguagem será imperativa, funcional, procedimental etc. e que tipo de comandos, funções e instruções terá. Depois, é preciso decidir se será uma linguagem *interpretada*, *compilada*, ou ambas.

Finalmente, escreve-se o programa — geralmente em Assembler, C, ou qualquer outra linguagem otimizada para o desenvolvimento de ferramentas de software — que irá realizar o processo de tradução da nova linguagem em códigos de máquina destinados ao computador escolhido.

As técnicas para a elaboração do compilador ou interpretador, complexas e altamente especializadas, são ensinadas em cursos superiores de análise de sistemas.

```
ESCREVA PRIMEIRO FELINOS
```

Isso fará com que o primeiro elemento da lista, que é TIGRE, seja impresso na tela.

Uma lista pode ter apenas um elemento, ou nenhum. Nesse caso, denomina-se *lista vazia*, e é representada pelo símbolo [ ]. Para unir duas listas, usamos o comando primitivo **SENTENCE** (ou **SENTENÇA**):

```
PRINT SENTENCE COLEGAS IMPARES
```

```
ESCREVA SENTENÇA COLEGAS
IMPARES
```

Procuramos aqui demonstrar a riqueza e a versatilidade do LOGO. Esperamos ter motivado o leitor a continuar explorando essa poderosa linguagem.

FUNÇÕES PODEROSAS

- MAIOR MÚLTIPLO
- MENOR MÚLTIPLO
- RESTO DE UMA DIVISÃO
- ARREDONDAMENTO
- PAR OU ÍMPAR?

O comando DEF FN permite incorporar ao elenco de funções matemáticas do BASIC um conjunto adicional de cálculos de grande interesse para o programador. Utilize-o.

No artigo da página 608, vimos como definir novas funções e usá-las em diversos tipos de manipulação de dados. Trataremos aqui de certas funções matemáticas úteis ao programador.

Um extenso grupo de funções matemáticas refere-se aos *módulos*, números que "cabem" dentro de outros números, ou que restam de uma divisão.

Há várias aplicações para esses cálculos. Tomemos como exemplo uma questão corriqueira, que envolve a categoria mais geral dos múltiplos: qual é o primeiro múltiplo de 100, menor ou igual a 345? Não é preciso ser nenhum gênio para chegar à resposta — 300, que é o *arredondamento* do número 345 para baixo, até a centena mais próxima. Mas qual é o primeiro múltiplo de 64, menor ou igual a 511? Eis uma função que resolve esse problema:

```
DEF FNMN(N1,N2)=INT(N1/N2)*N2
```

O segundo argumento corresponde ao número cujo múltiplo queremos achar; o primeiro, ao valor máximo desse múltiplo. Nos exemplos dados, poderíamos usar: **PRINT FNMN(345,100)** e **FNMN(511,64)**. Como o Apple, o TK-2000 e o TRS-Color não admitem mais que um argumento por função, a linha anterior não pode ser executada nessas máquinas. A solução é utilizar só um argumento (veja o artigo da página 608).

Também interessante é a função que determina o primeiro múltiplo de um número, superior a um certo valor. Exemplo: o primeiro múltiplo de 100 maior que 3 022 é 3 100.

```
DEF FNMM(N1,N2)=INT(N1/N2)*N2+
    N2
```

Os argumentos têm o mesmo significado que os da função anterior.

Existem ainda outros tipos de arredondamento. O mais comum consiste em arredondar os dígitos da parte fracionária de um número usando a "regra do 5": se o dígito a desprezar é maior ou igual a 5, o que está à sua esquerda é arredondado para cima; caso contrário, este não muda. A função que executa esse tipo de arredondamento é:

```
DEF FNAR(N,D)=INT(N*10**D)/10**
D
```

```
DEF FNAR(N,D)=INT(N*10↑D)/10↑D
```

```
DEF FNAR(N,D)=INT(N*10^D)/10^D
```

A função tem dois argumentos: **N**, o número a arredondar, e **D**, o número de decimais desejado. Se quisermos, por exemplo, arredondar para dois decimais o número 23.4567, especificamos **PRINT FNAR(23.4567,2)**.

Também é útil a função de arredondamento que determina o próximo número inteiro a partir de um número fracionário. É uma função diferente de **INT** (que obtém a parte inteira de um número, arredondando em alguns casos), ou de **FIX** (que determina apenas a parte inteira, sem arredondar):

```
DEF FNRD(N)=FIX((FIX(N*10)+SGN
    (N)*5)/10)
```

```
DEF FNRD(N)=INT((INT(N*10)+SGN
    (N)*5)/10)
```

Outra função importante é a que calcula o resto inteiro de uma divisão entre dois números **N1** e **N2**.

```
DEF FNRE(N1,N2)=N1-FNMN(N1,N2)
```

Utilizamos aqui a função **FNMN**, já definida. A função **FNRE** pode ser empregada quando se quer determinar se um número é par ou ímpar.

```
DEF FNPI(N)=2-INT(N/2)*2
```

```
DEF FNPI(N)=FNRE(N,2)
```

Se o resultado da função for 1, o número é ímpar; se for 0, é par.

O menu deste programa permite testar as funções apresentadas:

```
25 DEF FNMM(N1,N2)=INT(N1/N2)*N
2+N2
30 DEF FNMN(N1,N2)=INT(N1/N2)*N
2
35 DEF FNRE(N1,N2)=N1-FNMN(N1,N
2)
40 DEF FNRD(N)=FIX((FIX(N*10)+S
GN(N)*5)/10)
50 DEF FNPI(N)=FNRE(N,2)
60 DEF FNAR(N,D)=INT(N*10^D)/10
^D
100 CLS
110 PRINT "DEMONSTRACAO DE FUNC
OES MATEMATICAS"
120 PRINT
130 PRINT "(1)   PRIMEIRO MULTI
PLO MAIOR"
140 PRINT "(2)   PRIMEIRO MULTI
PLO MENOR OU IGUAL"
150 PRINT "(3)   RESTO DE UMA D
IVISAO"
160 PRINT "(4)   ARREDONDAMENTO
 DE UM NUMERO"
165 PRINT "(5)   ARREDONDAMENTO
PARA O MAIOR INTEIRO"
170 PRINT "(6)   PAR OU IMPAR"
180 PRINT "(7)   FIM"
190 PRINT
200 INPUT "OPCAO ";OP
210 IF OP=7 THEN STOP
220 IF OP>3 THEN INPUT "ARGUMEN
TO ";N:GOTO 250
230 INPUT "PRIMEIRO ARGUMENTO "
;N1
235 INPUT "SEGUNDO ARGUMENTO ";
N2
240 IF OP=1 THEN X=FNMM(N1,N2)
250 IF OP=2 THEN X=FNMN(N1,N2)
260 IF OP=3 THEN X=FNRE(N1,N2)
270 IF OP=4 THEN X=FNAR(N)
275 IF OP=5 THEN X=FNRD(N)
280 IF OP=6 THEN X=FNPI(N)
290 PRINT "RESULTADO = ";X
300 GOTO 120
```

PAPEL, PEDRA, TESOURA

Quer dizer que você acredita ser fácil blefar um computador? Experimente estes programas: eles mudarão sua "arrogante" opinião, colocando-o diante de um imbatível adversário.

- JOGOS DE BLEFE
- ALEATORIEDADE
- COMO CLASSIFICAR AS POSSIBILIDADES
- AS REGRAS DO JOGO
- O USO DA ESTATÍSTICA
- O QUE É ALISAMENTO EXPONENCIAL
- TÉCNICA DE SOMA CUMULATIVA

Um computador que possa enfrentar um blefe não é coisa do futuro: já existe aqui e agora. Com os programas em BASIC deste artigo, não há escapatória: a máquina será implacável com quem ousar desafiá-la em um clássico jogo de blefe: *Papel, Pedra, Tesoura*.

É UMA PEDRADA

Jogos de blefe são todos aqueles em que ambos os participantes mostram suas jogadas ao mesmo tempo, não havendo nenhuma informação prévia sobre a probabilidade da jogada de cada um. Um exemplo clássico é *Papel, Pedra, Tesoura*, no qual dois jogadores representam com gestos a forma de uma pedra, de uma folha de papel ou de uma tesoura. A escolha de cada um é revelada simultaneamente, como em um jogo de par ou ímpar. Ganha a rodada aquele que mostrar o objeto mais poderoso, de acordo com as seguintes regras:

- tesoura corta papel, portanto, tesoura ganha;
- papel embrulha pedra, portanto, papel ganha;
- pedra quebra tesoura, portanto, pedra ganha.

Se ambos os jogadores escolherem o mesmo objeto, haverá empate. Joga-se um número predeterminado de rodadas — vinte, por exemplo — e conta-se um ponto para cada rodada ganha.

Neste artigo, desenvolveremos duas versões de um programa que joga *Papel, Pedra, Tesoura* muito bem. Tão bem que você até se sentirá um pouco abalado quando perceber que o computador parece adivinhar suas intenções...

MAIS SOBRE BLEFE

Como um programa é capaz de predizer eventos aparentemente aleatórios? Uma das "saídas" seria a simples adivinhação — ou seja, o programa sortearia números ao acaso (digamos, o número 1 representaria o papel, o 2, a pedra, e o 3, a tesoura), tanto para determinar a jogada do computador, quanto para tentar adivinhar a do oponente. Um jogo como este não é satisfatório do ponto de vista estratégico: com um número de rodadas grande, certamente se chega a um empate.

Para sorte do computador, entretanto, o homem não age de maneira puramente aleatória, tendendo a exibir com maior freqüência um ou dois tipos de objetos. Quase sempre, porém, os padrões de apresentação são bem mais complexos, seguindo ciclos ou mudando gradativamente as probabilidades.

Além de sua incapacidade natural de atuar como um gerador perfeito de números aleatórios, o jogador humano tem mais um ponto vulnerável: a compulsão a responder ao que acontece no jogo. Quer esteja ganhando, quer esteja perdendo, é quase certo que formulará "teorias" — apelando, via de regra, para a superstição — sobre quais são as respostas com maior chance de garantir-lhe a vitória. A distração proporcionada pelo jogo também tende a reduzir a frieza matemática de um jogador, mesmo que ele lute contra isso.

De qualquer forma, não resta dúvida de que um jogador humano, por ser humano, tentará ampliar seus ganhos ao máximo de alguma estratégia. Isso significa que haverá sempre um viés, ou vício de jogada, que, se cuidadosamente analisado, poderá dar indícios de como será seu próximo lance.

Essa análise poderia ser feita por meio de cálculos estatísticos de diversos tipos. Mas o computador é bem mais eficiente no que se refere a cálculos — e é isto o que lhe permitirá sair vitorioso em um jogo de blefe.

Se um jogador modifica sua estratégia gradualmente, a técnica estatística mais adequada é a do *alisamento exponencial*. Se, ao contrário, o oponente muda de tática rapidamente, a melhor técnica é a da *soma cumulativa*.

COMO JOGAR

Para enfrentar o computador em *Papel, Pedra, Tesoura*, você precisa simplesmente pressionar a tecla 1, 2 ou 3, quando chegar a sua vez.

É importante notar o seguinte: em-

bora a máquina tenha uma grande chance de vencê-lo, ela não frauda o jogo, "olhando" sua jogada antes de decidir a dela. Na verdade, o método utilizado consiste em fazer o computador analisar as escolhas já feitas e decidir *antes* que você entre uma nova jogada.

ALISAMENTO EXPONENCIAL

O alisamento exponencial é a técnica estatística empregada para regularizar o contorno de uma curva cheia de altos e baixos. Se você colocar em um gráfico o número de pedras, tesouras e papéis que o oponente apresentou em cada unidade de tempo (por exemplo, a cada dez jogadas), a curva terá o aspecto de uma serra. Porém, regularizando o contorno da curva — ou seja, tornando as subidas e descidas mais "suaves" —, o computador detectará alguma tendência significativa a longo prazo, como, por exemplo, um aumento expressivo na apresentação de tesouras.

As técnicas de alisamento utilizam somas ponderadas das jogadas anteriores. No caso específico do alisamento exponencial, pesos menores são dados para jogadas mais antigas, e pesos maiores, para jogadas mais recentes.

Ao executar o programa, ele inicialmente pedirá que você entre um "fator de esquecimento". Quanto maior o valor (inteiro, positivo) que você fornecer, maior o número de jogadas anteriores que o computador deixará de considerar. Assim, ficará mais fácil vencê-lo.

```
5 CLEAR 31999: GOSUB 500:
BORDER 0: PAPER 0: INK 7:
CLS
7 DIM A$(3,9): LET A$(1)="PE
DRA": LET A$(2)="PAPEL": LET
A$(3)="TESOURA"
10 DIM H(3,2): DIM X(2): DIM
Q(3,3): DIM C(2,3): DIM A(2,3
): DIM P(3,3): CLS
20 FOR I=1 TO 3: LET H(I,1)=-
COS ((I-2)*PI*2/3): LET H(I,2
)=-SIN ((I-2)*PI*2/3): NEXT I
30 FOR T=1 TO 3: FOR J=1 TO 2
: LET A(J,T)=.01: NEXT J:
NEXT T
40 INPUT "DIGITE FATOR DE ESQ
UECIMENTO    (0-1). VALOR SUG
ERIDO  .85 ";W
70 FOR I=1 TO 3: FOR J=1 TO 3
: LET P(I,J)=SGN (I-J-3*INT (
(I-J+1.5)/3)): NEXT J: NEXT I
: LET S=0
80 LET U=1: LET S=0: LET U2=0
: LET WW=0: LET U3=0
100 LET V=INT (RND*3)+1
110 PRINT  INK 6; PAPER 2;"  1
=PEDRA, 2=PAPEL, 3=TESOURA  "
120 INK 6: PLOT 8,167: DRAW
239,0: DRAW 0,-159: DRAW -239,
0: DRAW 0,159
200 FOR T=1 TO 3: LET I=U
210 IF (U2=0 AND T=2) OR (U3=0
AND T=3) THEN  GOTO 280
220 IF T=2 THEN  LET I=ABS ((U
=U2)-2*(U=VV)-3*((U<>U2) AND (
U<>VV)))
230 IF T=3 THEN  LET I=ABS ((U
=U3)-2*(U=V3)-3*((U<>U3) AND (
U<>V3)))
240 FOR J=1 TO 2: LET A(J,T)=A
(J,T)*W+H(I,J): LET C(J,T)=C(J
,T)*W+3*H(I,J)*H(I,J)+.01
250 LET X(J)=A(J,T)/C(J,T):
NEXT J
260 FOR I=1 TO 3: LET Q(I,T)=1
/3: FOR K=1 TO 2: LET Q(I,T)=X
(K)*H(I,K): NEXT K: NEXT I
280 NEXT T: LET U2=U: LET VV=V
: IF U=V THEN  LET VV=U+1-3*
INT (U/3)
290 IF U<>V THEN  LET U3=U:
LET V3=V: IF P(U3,V3)<0 THEN
LET WW=U3: LET U3=V3: LET V3=
WW
300 LET X=-1E30: FOR T=1 TO 3:
IF (T=2 AND U2=0) OR (T=3 AND
U3=0) THEN  GOTO 370
310 IF T=1 THEN  GOTO 350
311 IF T=2 THEN  GOTO 320
312 IF T=3 THEN  GOTO 340
320 LET WW=6-U2-VV: LET Q1=Q(1
,T): LET Q2=Q(2,T): LET Q3=Q(3
,T): LET Q(U2,T)=Q1: LET Q(VV,
T)=Q2: LET Q(WW,T)=Q3
330 GOTO 350
340 LET WW=6-U3-V3: LET Q1=Q(1
,T): LET Q2=Q(2,T): LET Q3=Q(3
 T): LET Q(U3,T)=Q1: LET Q(V3,
T)=Q2: LET Q(WW,T)=Q3
350 FOR G=1 TO 3: LET P=0: FOR
I=1 TO 3: LET P=P+P(G,I)*Q(I,T
): NEXT I: IF P>X THEN  LET X=
P: LET V=G
360 NEXT G
370 NEXT T
400 INK 7: PRINT AT 2,4;"VOCE
DISSE "
405 FOR M=-3 TO 16: SOUND .01,
M: NEXT M
410 LET K$=INKEY$: IF K$=""
THEN  GOTO 410
412 IF K$<"1" OR K$>"3" THEN
GOTO 410
415 LET U=VAL K$
420 PRINT AT 3,4;A$(U);AT 2,21
; INK 5;"EU DISSE ";AT 3,21;A$
(V)
430 POKE 23681,U-1: LET O=USR
32000: POKE 23681,127+V: LET O
=USR 32000
440 LET S=S+P(U,V): PRINT AT
16,8;"SEU SCORE E ";S;"  "
450 INVERSE 1: IF V=U THEN
PRINT AT 18,10;" UM EMPATE ":
GOTO 490
460 IF (U=3 AND V=2) OR (U=2
AND V=1) OR (U=1 AND V=3) THEN
 PRINT AT 18,12;" VOCE VENCEU
": GOTO 490
470 PRINT AT 18,11;" EU VENCI"
490 INVERSE 0: FOR D=1 TO 2:
PAUSE 0: NEXT D: FOR N=2 TO 18
: PRINT  PAPER 0; INK 7;AT N,2
;"
": NEXT N: GOTO 200
500 FOR N=32000 TO 32284: READ
A: PRINT N,A: NEXT N
505 LET N=32069: POKE 23728,N-
256*INT (N/256): POKE 23729,
INT (N/256)
510 RETURN
520 DATA 33,6,72,58,129,92,203
,127,40,5,203,191,33,22,72,221
,42,176,92,17,72,0,254,0,40
530 DATA 8,254,1,40,2,221,25,
221,25,221,229,209,6,3,197,229
,6,8,197,1,3,0,235,237,176,235
,36,1,3,0
540 DATA 237,66,193,16,239,225
,1,32,0,9,193,16,227,201
550 DATA 0,0,0,0,0,0,0,0,0,0,0
,0,0,30,0,0,97,128,3,129,192,4
,15,225,10,56,112,21,32,48,30,
192,16,11,160,16,7,120,32
560 DATA 3,181,96,1,234,192,0,
255,128,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
570 DATA 0,16,0,0,56,0,0,124,0
,0,254,0,1,223,0,3,111,128,7,
191,192,14,219,224,27,109,240,
53,254,248,106,219,112,245,109
,224,122
580 DATA 190,192,61,91,128,30,
239,0,15,182,0,7,92,0,2,232,0,
1,240,0,0,224,0,0,0,0
590 DATA 0,0,0,0,0,0,0,0,0,0
600 DATA 0,0,0,0,0,0,0,0,0,56,0,
0,104,0,0,208,0,1,160,0,3,64,0
,6,128,0,13,0,0,26,0,0,52,0,1,
232,0,63,48,0
610 DATA 100,32,0,196,96,0,201
,192,0,115,96,0,6,32,0,4,32,0,
6,64,0,3,128,0,0,0,0,0,0,0,0,0
,0
```

```
5 CLEAR 1000:PMODE 3,1:COLOR 4,
2:PCLS
6 GOSUB 1000
7 C=8*ATN(1)/3
10 DIM H(3,2),X(2),Q(3,3),C(2,3
),A(2,3),P(3,3):CLS
20 FOR I=1 TO 3:H(I,1)=-COS((I-
2)*C):H(I,2)=-SIN((I-2)*C):NEXT
30 FOR T=1 TO 3:FOR J=1 TO 2:A(
J,T)=0:C(J,T)=.01:NEXT J,T
40 PRINT"DIGITE O FATOR DE ESQU
ECIMENTO. INTERVALO PERMITIDO
 0 A 1."
50 PRINT"0=SEM MEMORIA ALEM DE
1 JOGADA":PRINT"1=ESTRATEGIA FI
XA"
60 PRINT"VALOR SUGERIDO=.85":IN
PUT W
70 FOR I=1 TO 3:FOR J=1 TO 3:FO
R J=1 TO 3:P(I,J)=SGN(I-J-3*INT
((I-J+1.5)/3)):NEXT J,I
80 S=0:U2=0:WW=0:U3=0
100 V=RND(3)
110 COLOR 4:PCLS:LINE(8,20)-(24
```

```
7,171),PSET,B
120 DRAW"BM34,10S4C4":FOR K=1 T
O 3:DRAW N$(K)+E$+A$(K)+"BR4":N
EXT
140 GOTO 400
200 FOR T=1 TO 3:I=U
210 IF(U2=0 AND T=2) OR (U3=0 A
ND T=3) THEN 280
220 IF T=2 THEN I=-(U=U2)-2*(U=
VV)-3*((U<>U2)AND(U<>VV))
230 IF T=3 THEN I=-(U=U3)-2*(U=
V3)-3*((U<>U3)AND(U<>V3))
240 FOR J=1 TO 2:A(J,T)=A(J,T)*
W+H(I,J):C(J,T)=C(J,T)*W+3*H(I,
J)*H(I,J)+.01
250 X(J)=A(J,T)/C(J,T):NEXT
260 FOR I=1 TO 3:Q(I,T)=1/3:FOR
 K=1 TO 2:Q(I,T)=Q(I,T)+X(K)*H(
I,K):NEXT K,I
280 NEXT:U2=U:VV=V:IF U=V THEN
VV=U+1-3*INT(U/3)
290 IF U<>V THEN U3=U:V3=V:IF P
(U3,V3)<0 THEN WW=U3:U3=V3:V3=W
W
300 X=-1E30:FOR T=1 TO 3:IF(T=2
 AND U2=0) OR (T=3 AND U3=0) TH
EN 370
310 ON T GOTO 350,320,340
320 WW=6-U2-VV:Q1=Q(1,T):Q2=Q(2
,T):Q3=Q(3,T):Q(U2,T)=Q1:Q(VV,T
)=Q2:Q(WW,T)=Q3
330 GOTO 350
340 WW=6-U3-V3:Q1=Q(1,T):Q2=Q(2
,T):Q3=Q(3,T):Q(U3,T)=Q1:Q(V3,T
)=Q2:Q(WW,T)=Q3
350 FOR G=1 TO 3:P=0:FOR I=1 TO
 3:P=P+P(G,I)*Q(I,T):NEXT:IF P>
X THEN X=P:V=G
360 NEXT G
370 NEXT T
400 SCREEN 1,0:DRAW"BM12,30"+Y$
+S$
410 A$=INKEY$:IF A$<"1" OR A$>"
3" THEN 410
420 U=VAL(A$):DRAW A$(U)+"BM142
,30"+I$+S$+A$(V)
430 X=44:Y=90:ON U GOSUB 600,61
0,620
435 X=170:ON V GOSUB 700,710,72
0
440 LINE(80,175)-(240,190),PRES
ET,BF:S=S+P(U,V):DRAW"BM80,180B
D4R5U2L4U2R4BR8L4D4R4BR4U4R4D4N
L4BR4U4R4D2L2DFBR9L4U2NR4U2R4BR
8BU2S8":GOSUB 800:DRAW"S4"
450 IF U=V THEN DRAW"BM100,160C
3"+D$:GOTO 490
460 IF(U=3 AND V=2) OR (U2 AND
V=1) OR (U=1 AND V=3) THEN DRAW
"BM30,160C3"+Y$+W$:GOTO 490
470 DRAW"BM160,160C1"+I$+W$
490 FOR K=1 TO 1000:NEXT:LINE(1
0,22)-(245,169),PRESET,BF
500 GOTO 200
600 PUT(X,Y)-(X+40,Y+38),PA,PSE
T:RETURN
610 PUT(X,Y)-(X+40,Y+38),SC,PSE
T:RETURN
620 PUT(X,Y)-(X+40,Y+38),ST,PSE
T:RETURN
700 PUT(X,Y)-(X+40,Y+38),PA,AND
:RETURN
710 PUT(X,Y)-(X+40,Y+38),SC,AND
:RETURN
720 PUT(X,Y)-(X+40,Y+38),ST,AND
:RETURN
800 FOR K=1 TO LEN(STR$(S))
810 B$=MID$(STR$(S),K,1):IF B$=
"-" THEN DRAW"BF2R4BE2":GOTO 83
0
820 IF B$<"0" OR B$>"9" THEN 83
0
825 DRAW N$(VAL(B$))
830 NEXT:RETURN
1000 FOR I=0 TO 9:READ N$(I):NE
XT
1010 DATA NR2D4R2U4BR2,BDEND4BR
2,R2D2L2D2R2BU4BR2,NR2BD2NR2BD2
R2U4BR2,D2R2D2U4BR2,NR2D2R2D2L2
BE4,D4R2U2L2BE2BR2,R2ND4BR2,NR2
D4R2U2NL2U2BR2,NR2D2R2D2U4BR2
1015 DIM PA(39),SC(39),ST(39)
1020 DRAW"BM0,22C3M22,0M40,18M2
0,38L4M0,22":PAINT(20,20)
1030 DRAW"BM32,20C2S8H4BH2HBG2F
4BFF3BL4H2BHH2BL3F2BF2F3BD3H2BH
3H2"
1040 GET(0,0)-(40,38),PA,G
1050 DRAW"BM50,30C3URURUR5S4UNR
6US8R3URNE8UE7R2G2DG6LD2LD2FDGL
GL2ULUEERE2L2DBM-3,-2D2G2L2U2"
1060 GET(50,0)-(90,38),SC,G
1070 DRAW"BM128,6L3GLGL2G3DF5R8
E2UEU3H3LU":PAINT(124,16)
1080 DRAW"BM128,8C2L3GLGNL2DFRU
ERER2U":PAINT(122,10):DRAW"BM13
4,14L3GLNG3D2RNF2R4NG2EUH":PAIN
T(130,18)
1090 GET(100,0)-(140,38),ST,G
1100 E$="BR2BDNR3BD2R3BE3BR"
1102 A$(1)="NR2D4U2R4U2BF4U2NR2
U2R4D4BR4U2NU2R5U2NL2BR8L4D2NR2
D2R4BR3U4R4D2LDFRBE4"
1104 A$(2)="BD4R4U2L3U2R4BR8L4D
4R4BR4U4BR8L4D2R3D2L3BR7R4U2L3U
2R4BR3ND4R5D4NL2BR3U4R4D2LDFBR5
R3U2L3U2R4BR2"
1110 A$(3)="BD4R4U2L3U2R4BR3R3N
D4R2BR3NR5D4R5U4BR4ND4F3RFU4BF4
NR4U2NR2U2R5BR3"
1120 I$="ND4BR6":Y$="C4F2ND2RE2
BR3D4R4U4LBR6D4R3U4BR6"
1130 W$="D4RERERFRFU4BR4ND4BR5N
D4F2RF2U4"
1140 S$="BD4R4U2L3U2R4BR3ND4R4D
2NLD2BR4U4BR4D4R3EU2HBR9"
1150 D$="ND4R4D2NLD2BR7U4R2FD2G
BR5U4R4D2LDFBR4U4R4D2LF2BR3NU4E
RERFRFU4"
1200 RETURN
```

Nas versões para o MSX, o Apple e o TK-2000 não há apresentação de gráficos.

```
6 GOSUB 1000
7 C=8*ATN(1)/3:NU=0:NV=0
10 DIM H(3,2),X(2),Q(3,3),C(2,3
),A(2,3),P(3,3)
20 FOR I=1 TO 3
25 H(I,1)=-COS((I-2)*C):H(I,2)
=-SIN((I-2)*C)
27 NEXT I
30 FOR T=1 TO 3:FOR J=1 TO 2:A(
J,T)=0:C(J,T)=0.01:NEXT J:NEXT
T
35 LOCATE 0,4
40 PRINT "FATOR DE ESQUECIMENTO
 A SER USADO"
45 PRINT:PRINT "VALOR ENTRE 0 E
 1 :"
50 PRINT "0 = MEMORIA DE 1 JOGO
 APENAS":PRINT "1 = ESTRATEGIA
INVARIAVEL"
60 PRINT "VALOR SUGERIDO = 0.85
"
65 PRINT:PRINT "VALOR ";:INPUT
W
67 FOR I=4 TO 11:LOCATE 0,I
68 PRINT "
                "
69 NEXT I
70 FOR I=1 TO 3:FOR J=1 TO 3
72 P(I,J)=1
75 IF I-J-3*INT((I-J+1.5)/3)<0
THEN P(I,J)=-1
77 NEXT J:NEXT I
80 S=0:U2=0:WW=0:U3=0
100 V=INT(3*RND(1))+1
120 LOCATE 3,4:PRINT "VOCE"
130 LOCATE 20,4:PRINT "EU"
140 GOTO 400
200 FOR T=1 TO 3:I=U
210 IF (U2=0 AND T=2) OR (U3=0
AND T=3) THEN 270
220 IF T=2 THEN I=-(U=U2)-2*(U=
VV)-3*((U<>U2) AND (U<>VV))
230 IF T=3 THEN I=-(U=U3)-2*(U=
V3)-3*((U<>U3) AND (U<>V3))
240 FOR J=1 TO 2:LET A(J,T)=A(J
,T)*W+H(I,J)
245 C(J,T)=C(J,T)*W+3*H(I,J)*
H(I,J)+0.01
250 LET X(J)=A(J,T)/C(J,T):NEXT
 J
260 FOR I=1 TO 3:Q(I,T)=1/3:FOR
 K=1 TO 2
265 Q(I,T)=Q(I,T)+X(K)*H(I,K):
NEXT K:NEXT I
270 NEXT T
280 U2=U:VV=V
285 IF U=V THEN VV=U+1-3*INT(U/
3)
290 IF U<>V THEN U3=U:V3=V:IF P
(U3,V3)<0 THEN WW=U3:U3=V3:V3=W
W
300 X=-9999999:FOR T=1 TO 3
305 IF (T=2 AND U2=0) OR (T=3 A
ND U3=0) THEN 370
310 ON T GOTO 350,320,340
320 WW=6-U2-VV:Q1=Q(1,T):Q2=
Q(2,T):Q3=Q(3,T)
325 Q(U2,T)=Q1:Q(VV,T)=Q2:Q(WW
,T)=Q3
330 GOTO 350
340 WW=6-U3-V3:Q1=Q(1,T):Q2=Q(
2,T):Q3=Q(3,T)
345 Q(U3,T)=Q1:Q(V3,T)=Q2:Q(WW
,T)=Q3
350 FOR G=1 TO 3:P=0:FOR I=1 TO
 3
355 LET P=P+P(G,I)*Q(I,T):NEXT
I
357 IF P>X THEN X=P:V=G
360 NEXT G
370 NEXT T
```

```
400 LOCATE 0,21:PRINT "JOGUE: 1
=PAPEL 2=TESOURA 3=PEDRA ";
410 A$=INKEY$:IF A$<"1" OR A$>"
3" THEN 410
420 U=VAL(A$)
425 LOCATE 3,10:PRINT A$(U)
430 LOCATE 20,10:PRINT A$(V)
445 S=S+P(U,V)
447 LOCATE 12,16:PRINT "
      ":LOCATE 12,16
450 IF U=V THEN PRINT "EMPATE":
GOTO 490
460 IF (U=3 AND V=2) OR (U=2 AN
D V=1) OR (U=1 AND V=3) THEN PR
INT "VOCE GANHOU !":NU=NU+1:GOT
O 490
470 PRINT "GANHEI !":NV=NV+1
490 LOCATE 3,18:PRINT NU:LOCATE
 20,18:PRINT NV
500 GOTO 200
1000 LET A$(1)="PAPEL  ":A$(2)=
"TESOURA":A$(3)="PEDRA  "
1010 CLS
1020 PRINT "P A P E L  -  P E D
 R A - T E S O U R A"
1030 PRINT "==================
==================="
1040 LOCATE 0,20
1050 PRINT "==================
==================="
1200 RETURN
```

```
6 GOSUB 1000
7 LET C=8*ATN(1)/3:NU=0:NV=0
10 DIM H(3,2),X(2),Q(3,3),C(2,3
   ),A(2,3),P(3,3)
20 FOR I=1 TO 3
25 LET H(I,1)=-COS((I-2)*C):H(I
   ,2)=-SIN((I-2)*C)
27 NEXT I
30 FOR T=1 TO 3:FOR J=1 TO 2:A(
   J,T)=0:C(J,T)=0.01:NEXT J:
   NEXT T
35 VTAB 4:HTAB 1
40 PRINT "FATOR DE ESQUECIMENTO
   A SER USADO"
45 PRINT:PRINT "VALOR ENTRE 0 E
   1 :"
50 PRINT "0 = MEMORIA DE 1 JOGO
   APENAS":PRINT "1 = ESTRATEGI
   A INVARIAVEL"
60 PRINT "VALOR SUGERIDO = 0.85
   "
65 PRINT:PRINT "VALOR ";:INPUT
   W
67 FOR I=4 TO 11:VTAB I:HTAB 1
68 PRINT "
                "
69 NEXT I
70 FOR I=1 TO 3:FOR J=1 TO 3
72 LET P(I,J)=1
75 IF I-J-3*INT((I-J+1.5)/3)<0
   THEN P(I,J)=-1
77 NEXT J:NEXT I
80 LET S=0:U2=0:WW=0:U3=0
100 LET V=INT(3*RND(1))+1
120 VTAB 4:HTAB 3:PRINT "VOCE"
130 VTAB 4:HTAB 20:PRINT "EU"
140 GOTO 400
200 FOR T=1 TO 3:I=U
210 IF (U2=0 AND T=2) OR (U3=0
    AND T=3) THEN 270
220 IF T=2 THEN I=-(U=U2)-2*(U=
    VV)-3*((U<>U2) AND (U<>VV))
230 IF T=3 THEN I=-(U=U3)-2*(U=
    V3)-3*((U<>U3) AND (U<>V3))
240 FOR J=1 TO 2:LET A(J,T)=A(J
    ,T)*W+H(I,J)
245 LET C(J,T)=C(J,T)*W+3*H(I,J
    )*H(I,J)+0.01
250 LET X(J)=A(J,T)/C(J,T):NEXT
    J
260 FOR I=1 TO 3:Q(I,T)=1/3:FOR
    K=1 TO 2
265 LET Q(I,T)=Q(I,T)+X(K)*H(I,
    K):NEXT K:NEXT I
270 NEXT T
280 LET U2=U:VV=V
285 IF U=V THEN VV=U+1-3*INT(U/
    3)
290 IF U<>V THEN U3=U:V3=V:IF P
    (U3,V3)<0 THEN WW=U3 : U3 =
    V3:V3=WW
300 LET X=-9999999:FOR T=1 TO 3
305 IF (T=2 AND U2=0) OR (T=3 A
    ND U3=0) THEN 370
310 ON T GOTO 350,320,340
320 LET WW=6-U2-VV:Q1=Q(1,T):Q2
    =Q(2,T):Q3=Q(3,T)
325 LET Q(U2,T)=Q1:Q(VV,T)=Q2:Q
    (WW,T)=Q3
330 GOTO 350
340 LET WW=6-U3-V3:Q1=Q(1,T):Q2
    =Q(2,T):Q3=Q(3,T)
345 LET Q(U3,T)=Q1:Q(V3,T)=Q2:Q
    (WW,T)=Q3
350 FOR G=1 TO 3:P=0:FOR I=1 TO
    3
355 LET P=P+P(G,I)*Q(I,T):NEXT
    I
357 IF P>X THEN X=P:V=G
360 NEXT G
370 NEXT T
400 VTAB 21:HTAB 1:PRINT"JOGUE:
    1=PAPEL 2=TESOURA 3=PEDRA";
410 GET A$:IF A$<"1" OR A$>"3"
    THEN 410
420 LET U=VAL(A$)
425 VTAB 10:HTAB 3:PRINT A$(U)
430 VTAB 10:HTAB 20:PRINT A$(V)
445 LET S=S+P(U,V)
447 VTAB 16:HTAB 12:PRINT "
          ":VTAB 16:HTAB 12
450 IF U=V THEN PRINT "EMPATE":
    GOTO 490
460 IF (U=3 AND V=2) OR (U=2 AN
    D V=1) OR (U=1 AND V=3)THEN
    PRINT "VOCE GANHOU !":NU=NU
    +1:GOTO 490
470 PRINT "GANHEI !":NV=NV+1
490 VTAB 18:HTAB 3:PRINT NU
    VTAB 18:HTAB 20:PRINT NV
500 GOTO 200
1000 LET A$(1)="PAPEL  ":A$(2)=
     "TESOURA":A$(3)="PEDRA  "
1010 HOME
1020 PRINT "P A P E L  -  P E D
     R A - T E S O U R A"
1030 PRINT "==================
     ==================="
1040 VTAB 20:HTAB 1
1050 PRINT "==================
     ==================="
1200 RETURN
```

SOMA CUMULATIVA

Soma cumulativa é a técnica usada para analisar o desempenho do oponente num período de tempo em que se registram mudanças bruscas de respostas.

Os gráficos empregados pelo programa são iguais aos do programa anterior (apenas nas versões para o TRS-Color e Spectrum. Nas versões para o MSX, Apple e TK-2000, a parte comum entre os dois programas se refere à apresentação dos resultados). Assim, as linhas de programa correspondentes à seção relativa à apresentação dos resultados não mudam, devendo ser incorporadas ao novo programa como estão.

Como no programa anterior, o jogador deve entrar uma informação que regula a tática do computador (um número entre 1 e 9999).

S

Use as linhas 520 a 610 do programa anterior e acrescente estas:

```
 5 CLEAR 31999: GOSUB 500:
 BORDER 0: PAPER 0: INK 7:
 CLS
 7 DIM A$(3,9): LET A$(1)="PE
 DRA": LET A$(2)="PAPEL": LET
 A$(3)="TESOURA"
10 DIM B(3): DIM U(3): DIM S(
3): DIM H(3,2): DIM X(3): DIM
Q(3): DIM P(3,3): DIM M(3)
15 LET MM=60: DIM A(3,MM):
DIM Z(MM)
20 FOR I=1 TO 3: LET H(I,1)=-
COS ((I-2)*PI*2/3): LET H(I,2
)=-SIN ((I-2)*PI*2/3): NEXT I
30 LET M(1)=1: LET M(2)=0:
LET M(3)=0
40 INPUT "DIGITE RAZAO DE SEM
ELHANCA PARA CADA JOGADA (1 A
 9999), 1=NOVA  ESTRATEGIA PA
RA CADA JOGADA,    9999=MESMA
 ESTRATEGIA. ";W
70 FOR I=1 TO 3: FOR J=1 TO 3
: LET P(I,J)=SGN (I-J-3*INT (
(I-J+1.5)/3)): NEXT J: NEXT I
: LET S=0
80 LET NN=0: LET U=1: LET S=0
: LET U2=0: LET WW=0: LET U3=0
100 LET V=INT (RND*3)+1
110 PRINT  INK 6; PAPER 2;"  1
=PEDRA, 2=PAPEL, 3=TESOURA  "
120 INK 6: PLOT 8,167: DRAW
239,0: DRAW 0,-159: DRAW -239,
0: DRAW 0,159
130 GOTO 400
200 LET Y=-1E30: LET Z=-1E30:
FOR T=1 TO 3: LET I=U: IF T=1
THEN  GOTO 230
210 IF (U2=0 AND T=2) OR (U3=0
AND T=3) THEN  GOTO 264
220 IF T=2 THEN  LET I=ABS ((U
=U2)-2*(U=VV)-3*((U<>U2) AND (
U<>VV)))
```

```
230 IF T=3 THEN  LET I=ABS ((U
=U3)-2*(U=V3)-3*((U<>U3) AND (
U<>V3)))
240 LET A(T,M(T))=I
250 FOR J=1 TO 3: LET B(J)=0:
NEXT J: LET N=0: LET N2=M(1)
252 FOR M=M(T) TO 1 STEP -1:
LET B(A(T,M))=B(A(T,M))+1: LET
N=N+1: LET N2=N2-1
256 FOR J=1 TO 3: LET Q(J)=B(J
)/N: LET X(J)=Q(J)+(Q(J)=0):
NEXT J
258 LET Q=B(1)*LN (X(1))+B(2)*
LN (X(2))+B(3)*LN (X(3)): IF
NN>1 AND N2<>0 THEN  LET Q=Q+Z
(N2)
260 IF Q>Z THEN  LET Z=Q: LET
S(1)=Q(1): LET S(2)=Q(2): LET
S(3)=Q(3): LET SS=T: LET NN=N2
262 NEXT M: IF Q>Y THEN  LET Y
=Q: LET TT=T: LET U(1)=Q(1):
LET U(1)=Q(2): LET U(3)=Q(3)
264 NEXT T: LET T=TT: LET Z(M(
1))=Y: LET Q(1)=U(1): LET Q(2)
=U(2): LET Q(3)=U(3)
270 LET U2=U: LET VV=V: IF U2=
VV THEN  LET VV=U2+1-3*INT (U2
/3)
272 IF U<>V THEN  LET U3=U:
LET V3=V: IF P(U3,V3)<0 THEN
LET WW=U3: LET U3=V3: LET V3=
WW
274 LET M(1)=M(1)+1: LET M(2)=
M(2)+(U2>0): LET M(3)=M(3)=(U3
>0)
280 IF Q>Z-LN (W) THEN  GOTO
300
282 FOR T=1 TO 3: FOR M=NN+1
TO M(T): LET A(T,M-NN)=A(T,M):
NEXT M
284 LET M(T)=(NN-M(T))*(-1*((M
(T)>NN))): NEXT T
286 LET T=TT: LET Q(1)=S(1):
LET Q(2)=S(2): LET Q(3)=S(3)
290 FOR M=1 TO M(1)-1: LET Z(M
)=-1E30: NEXT M: LET Z(M(1))=Q
300 LET X=-1E30
310 IF T=1 THEN  GOTO 350
311 IF T=2 THEN  GOTO 320
312 IF T=3 THEN  GOTO 340
320 LET WW=6-U2-VV: LET Q1=Q(1
): LET Q2=Q(2): LET Q3=Q(3):
LET Q(U2)=Q1: LET Q(VV)=Q2:
LET Q(WW)=Q3
330 GOTO 350
340 LET WW=6-U3-V3: LET Q1=Q(1
): LET Q2=Q(2): LET Q3=Q(3):
LET Q(U3)=Q1: LET Q(V3)=Q2:
LET Q(WW)=Q3
350 FOR G=1 TO 3: LET P=0: FOR
I=1 TO 3: LET P=P+P(G,I)*Q(I):
NEXT I: IF P>X THEN  LET X=P:
LET V=G
360 NEXT G
400 INK 7: PRINT AT 2,4;"VOCE
DISSE "
405 FOR M=-3 TO 16: SOUND .01,
M: NEXT M
410 LET K$=INKEY$: IF K$=""
THEN  GOTO 410
412 IF K$<"1" OR K$>"3" THEN
GOTO 410
415 LET U=VAL K$
420 PRINT AT 3,4;A$(U);AT 2,21
; INK 5;"EU DISSE ";AT 3,21;A$
(V)
430 POKE 23681,U-1: LET O=USR
32000: POKE 23681,127+V: LET O
=USR 32000
440 LET S=S+P(U,V): PRINT AT
16,8;"SEU PLACAR E ";S;"  "
450 INVERSE 1: IF V=U THEN
PRINT AT 18,10;" UM EMPATE ":
GOTO 490
460 IF (U=3 AND V=2) OR (U=2
AND V=1) OR (U=1 AND V=3) THEN
 PRINT AT 18,10;" VOCE VENCEU
": GOTO 490
470 PRINT AT 18,10;"  EU VENCI
  "
490 INVERSE 0: FOR D=1 TO 2:
PAUSE 0: NEXT D: FOR N=2 TO 18
: PRINT  PAPER 0; INK 7;AT N,2
;"                              "
: NEXT N: GOTO 200
500 FOR N=32000 TO 32284: READ
A: POKE N,A: NEXT N
505 LET N=32069: POKE 23728,N-
256*INT (N/256): POKE 23729,
INT (N/256)
510 RETURN
```

Apague o programa anterior até a linha 435 e digite:

```
5 CLEAR 1000:PMODE 3,1:COLOR 4,
2:PCLS
6 GOSUB 1000
7 C=8*ATN(1)/3
10 DIM B(3),U(3),S(3),H(3,2),X(
3),Q(3),P(3,3),M(3)
15 MM=60:DIM A(3,MM),Z(MM)
20 FOR I=1 TO 3:H(I,1)=-COS((I-
2)*C):H(I,2)=-SIN((I-2)*C):NEXT
30 M(1)=1
40 CLS:INPUT"DIGITE RAZAO DE SE
MELHANCA PARA CADA JOGADA (1 A
9999),  1=NOVA ESTRATEGIA PARA
CADA JOGADA .   9999=MESMA ESTR
ATEGIA. ";W
70 FOR I=1 TO 3:FOR J=1 TO 3:P(
I,J)=SGN(I-J-3*INT((I-J+1.5)/3)
):NEXT J,I
80 U=1
100 V=RND(3)
110 COLOR 4:PCLS:LINE(8,20)-(24
7,171),PSET,B
120 DRAW"BM34,10S4C4":FOR K=1 T
O 3:DRAW N$(K)+E$+A$(K)+"BR4":N
EXT
140 GOTO 400
200 Y=-1E30:Z=-1E30:FOR T=1 TO
3:I=U:IF T=1 THEN 230
210 IF(U2=0 AND T=2)OR(U3=0 AND
 T=3) THEN 264
220 IF T=2 THEN I=-(U=U2)-2*(U=
VV)-3*((U<>U2)AND(U<>VV))
230 IF T=3 THEN I=-(U=U3)-2*(U=
V3)-3*((U<>U3)AND(U<>V3))
240 A(T,M(T))=I
250 FOR J=1 TO 3:B(J)=0:NEXT:N=
0:N2=M(1)
252 FOR M=M(T) TO 1 STEP -1:B(A
(T,M))=B(A(T,M))+1:N=N+1:N2=N2-
1
256 FOR J=1 TO 3:Q(J)=B(J)/N:X(
J)=Q(J)-(Q(J)=0):NEXT
258 Q=B(1)*LOG(X(1))+B(2)*LOG(X
(2))+B(3)*LOG(X(3)):IF NN>0 THE
N Q=Q+Z(N2)
260 IF Q>Z THEN Z=Q:S(1)=Q(1):S
(2)=Q(2):S(3)=Q(3):SS=T:NN=N2
262 NEXT M:IF Q>Y THEN Y=Q:TT=T
:U(1)=Q(1):U(2)=Q(2):U(3)=Q(3)
264 NEXT T:T=TT:Z(M(1))=Y:Q(1)=
U(1):Q(2)=U(2):Q(3)=U(3)
270 U2=U:VV=V:IF U2>VV THEN VV=
U2+1-3*INT(U2/3)
272 IF U<>V THEN U3=U:V3=V:IF P
(U3,V3)<0 THEN WW=U3:U3=V3:V3=W
W
274 M(1)=M(1)+1:M(2)=M(2)=(U2>0
):M(3)=M(3)-(U3>0)
280 IF Q>Z-LOG(W) THEN 300
282 FOR T=1 TO 3:FOR M=NN+1 TO
M(T):A(T,M-NN)=A(T,M):NEXT
284 M(T)=(NN-MT)*(M(T)>NN):NEXT
286 T=TT:Q(1)=S(1):Q(2)=S(2):Q(
3)=S(3)
290 FOR M=1 TO M(1)-1:Z(M)=-1E3
0:NEXT:Z(M(1))=Q
300 X=-1E30
310 ON T GOTO 350,320,340
320 WW=6-U2-VV:Q1=Q(1):Q2=Q(2):
Q3=Q(3):Q(U2)=Q1:Q(VV)=Q2:Q(WW)
=Q3
330 GOTO 350
340 WW=6-U3-V3:Q1=Q(1):Q2=Q(2):
Q3=Q(3):Q(U3)=Q1:Q(V3)=Q2:Q(WW)
=Q3
350 FOR G=1 TO 3:P=0:FOR I=1 TO
 3:P=P+P(G,I)*Q(I):NEXT:IF P>X
THEN X=P:V=G
360 NEXT
400 SCREEN 1,0:DRAW"BM12,30"+Y$
+S$
410 A$=INKEY$:IF A$<"1" OR A$>"
3" THEN 410
420 U=VAL(A$):DRAW A$(U)+"BM142
,30"+I$+S$+A$(V)
430 X=44:Y=90:ON U GOSUB 600,61
0,620
435 X=170:ON V GOSUB 700,710,72
0
```

Apague o programa anterior até a linha 390 e digite:

```
6 GOSUB 1000
7 C=8*ATN(1)/3:NU=0:NV=0
10 DIM B(3),U(3),S(3),H(3,2),X(
3),Q(3),P(3,3),M(3)
15 MM=60:DIM A(3,MM),Z(MM)
20 FOR I=1 TO 3
25 H(I,1)=-COS((I-2)*C):H(I,2)
=-SIN((I-2)*C)
27 NEXT I
30 M(1)=1
35 LOCATE 0,4
40 PRINT "FATOR DE VEROSSIMILHA
NÇA"
45 PRINT:PRINT "VALOR ENTRE 1 E
```

```
 9999 :"
50 PRINT "1 = NOVA ESTRATEGIA P
ARA CADA JOGO":PRINT "9999 = ME
SMA ESTRATEGIA"
65 PRINT:PRINT "VALOR ";:INPUT
W
67 FOR I=4 TO 10:LOCATE 0,I
68 PRINT "
                "
69 NEXT I
70 FOR I=1 TO 3:FOR J=1 TO 3
72 P(I,J)=1
75 IF I-J-3*INT((I-J+1.5)/3)<0
THEN P(I,J)=-1
77 NEXT J:NEXT I
80 U=1
100 V=INT(3*RND(1))+1
120 LOCATE 3,4:PRINT "VOCE"
130 LOCATE 20,4:PRINT "EU"
140 GOTO 400
200 Y=-99999:FOR T=1 TO 3:I=U
205 IF T=1 THEN 230
210 IF (U2=0 AND T=2) OR (U3=0
AND T=3) THEN 264
220 IF T=2 THEN I=-(U=U2)-2*(U=
VV)-3*((U<>U2) AND (U<>VV))
230 IF T=3 THEN I=-(U=U3)-2*(U=
V3)-3*((U<>U3) AND (U<>V3))
240 A(T,M(T))=I
250 FOR J=1 TO 3:B(J)=0:NEXT J:
N=0:N2=M(1)
252 FOR M=M(T) TO 1 STEP -1
254 B(A(T,M))=B(A(T,M))+1:N=N+1
:N2=N2-1
256 FOR J=1 TO 3:Q(J)=B(J)/N:X(
J)=Q(J)-(Q(J)=0):NEXT J
258 A=B(1)*LOG(X(1))+B(2)*LOG(X
(2))+B(3)*LOG(X(3))
259 IF NN>0 THEN Q=Q+Z(N2)
260 IF Q>Z THEN Z=Q:S(1)=Q(1):S
(2)=Q2:S(3)=Q3:SS=T:NN=N2
262 NEXT M
263 IF Q>Y THEN Y=Q:TT=T:U(1)=Q
(1):U(2)=Q(2):U(3)=Q(3)
264 NEXT T
265 T=TT:Z(M(1))=Y:Q(1)=U(1):
Q(2)=U(2):Q(3)=U(3)
270 U2=U:VV=V:IF U2>VV THEN
 VV=U2+1-3*INT(U2/3)
272 IF U<>V THEN U3=U:V3=V:IF P
(U3,V3)<0 THEN WW=U3:U3=V3:V3=W
W
274 M(1)=M(1)+1:LET M(2)=M(2)-(
U2>0):M(3)=M(3)-(U3>0)
280 IF Q>Z-LOG(W) THEN 300
282 FOR T=1 TO 3:FOR M=NN+1 TO
M(T)
283 A(T,M-NN)=A(T,M):NEXT M
284 M(T)=(NN-MT)*(M(T)>NN):
NEXT T
286 T=TT:Q(1)=S(1):Q(2)=S(2):
Q(3)=S(3)
290 FOR M=1 TO M(1)-1:Z(M)=
-99999:NEXT M
295 Z(M(1))=Q
300 X=-99999
310 ON T GOTO 350,320,340
320 WW=6-U2-VV:Q1=Q(1):Q2=Q(2):
Q3=Q(3)
325 Q(U2)=Q1:Q(VV)=Q2:Q(WW)=Q3
330 GOTO 350
340 WW=6-U3-V3:Q1=Q(1):Q2=Q(2):
Q3=Q(3)
345 Q(U3)=Q1:Q(V3)=Q2:Q(WW)=Q3
350 FOR G=1 TO 3:P=0:FOR I=1 TO
 3
355 P=P+P(G,I)*Q(I):NEXT I
357 IF P>X THEN X=P:V=G
360 NEXT G
```

Apague o programa anterior até a linha 390 e digite:

```
6 GOSUB 1000
7 LET C=8*ATN(1)/3:NU=0:NV=0
10 DIM B(3),U(3),S(3),H(3,2),X(
   3),Q(3),P(3,3),M(3)
15 LET MM=60:DIM A(3,MM),Z(MM)
20 FOR I=1 TO 3
25 LET H(I,1)=-COS((I-2)*C):H(I
   ,2)=-SIN((I-2)*C)
27 NEXT I
30 LET M(1)=1
35 VTAB 4:HTAB 1
40 PRINT "FATOR DE VEROSSIMILHA
   NCA"
45 PRINT:PRINT "VALOR ENTRE 1 E
    9999 :"
50 PRINT "1 = NOVA ESTRATEGIA P
   ARA CADA JOGO":PRINT "9999 =
    MESMA ESTRATEGIA"
65 PRINT:PRINT "VALOR ";:INPUT
   W
67 FOR I=4 TO 10:VTAB I:HTAB 1
68 PRINT "
                "
69 NEXT I
70 FOR I=1 TO 3:FOR J=1 TO 3
72 LET P(I,J)=1
75 IF I-J-3*INT((I-J+1.5)/3)<0
   THEN P(I,J)=-1
77 NEXT J:NEXT I
80 LET U=1
100 LET V=INT(3*RND(1))+1
120 VTAB 4:HTAB 3:PRINT "VOCE"
130 VTAB 4:HTAB 20:PRINT "EU"
140 GOTO 400
200 LET Y=-99999:FOR T=1 TO 3:I
   =U
205 IF T=1 THEN 230
210 IF (U2=0 AND T=2) OR (U3=0
   AND T=3) THEN 264
220 IF T=2 THEN I=-(U=U2)-2*(U=
   VV)-3*((U<>U2) AND (U<>VV))
230 IF T=3 THEN I=-(U=U3)-2*(U=
   V3)-3*((U<>U3) AND (U<>V3))
240 LET A(T,M(T))=I
250 FOR J=1 TO 3:B(J)=0:NEXT J:
   N=0:N2=M(1)
252 FOR M=M(T) TO 1 STEP -1
254 LET B(A(T,M))=B(A(T,M))+1:N
   =N+1:N2=N2-1
256 FOR J=1 TO 3:Q(J)=B(J)/N:X(
   J)=Q(J)-(Q(J)=0):NEXT J
258 LET A=B(1)*LOG(X(1))+B(2)*L
   OG(X(2))+B(3)*LOG(X(3))
259 IF NN>0 THEN Q=Q+Z(N2)
260 IF Q>Z THEN Z=Q:S(1)=Q(1):S
   (2)=Q2:S(3)=Q3:SS=T:NN=N2
262 NEXT M
263 IF Q>Y THEN Y=Q:TT=T:U(1)=Q
   (1):U(2)=Q(2):U(3)=Q(3)
264 NEXT T
```

COMO ENGANAR O COMPUTADOR

Se você não quiser ser derrotado pelo computador, precisará lançar mão de uma estratégia que neutralize ao máximo o método de análise utilizado no programa. A melhor alternativa, nesse caso, é empregar um processo absolutamente aleatório em suas jogadas. Para isso, rode um pequeno programa de sorteio de três números no seu micro, e imprima (ou então copie à mão) uma longa seqüência de jogadas. Seguindo-a à risca, o computador não terá nenhuma chance de prever suas jogadas. Em compensação, você também não conseguirá prever as jogadas do computador!

Outra técnica para enganar o computador consiste em mudar continuamente de estratégia. Os resultados, aqui, dependerão do programa que você estiver usando. Veja como o computador "enlouquece"...

```
265 LET T=TT:Z(M(1))=Y:Q(1)=U(1
   ):Q(2)=U(2):Q(3)=U(3)
270 LET U2=U:VV=V:IF U2>VV THEN
    VV=U2+1-3*INT(U2/3)
272 IF U<>V THEN U3=U:V3=V:IF P
   (U3,V3)<0 THEN WW=U3:U3=V3:
   V3=WW
274 LET M(1)=M(1)+1:LET M(2)=M(
   2)-(U2>0):M(3)=M(3)-(U3>0)
280 IF Q>Z-LOG(W) THEN 300
282 FOR T=1 TO 3:FOR M=NN+1 TO
   M(T)
283 LET A(T,M-NN)=A(T,M):NEXT M
284 LET M(T)=(NN-MT)*(M(T)>NN):
   NEXT T
286 LET T=TT:Q(1)=S(1):Q(2)=S(2
   ):Q(3)=S(3)
290 FOR M=1 TO M(1)-1:LET Z(M)=
   -99999:NEXT M
295 LET Z(M(1))=Q
300 LET X=-99999
310 ON T GOTO 350,320,340
320 LET WW=6-U2-VV:Q1=Q(1):Q2=Q
   (2):Q3=Q(3)
325 LET Q(U2)=Q1:Q(VV)=Q2:Q(WW)
   =Q3
330 GOTO 350
340 LET WW=6-U3-V3:Q1=Q(1):Q2=Q
   (2):Q3=Q(3)
345 LET Q(U3)=Q1:Q(V3)=Q2:Q(WW)
   =Q3
350 FOR G=1 TO 3:P=0:FOR I=1 TO
   3
355 LET P=P+P(G,I)*Q(I):NEXT I
357 IF P>X THEN X=P:V=G
360 NEXT G
```

A MATEMÁTICA DA IRREGULARIDADE (1)

Use seu micro na exploração das formas mais fascinantes da geometria de dimensões fracionadas, um instrumento da matemática que ajuda a explicar as irregularidades observadas na natureza.

Qual é o comprimento de um pedaço de corda? A resposta é muito fácil se a corda estiver esticada, pois bastará medi-la com uma fita métrica. Caso contrário, a tarefa será um pouco mais trabalhosa, pois teremos que medir um objeto irregular.

É evidente que, para obter as medidas de um objeto irregular, procuraremos estendê-lo, de modo a simplificar o trabalho. Porém, nem todas as coisas são suficientemente maleáveis para se fazer isso. Como agir então?

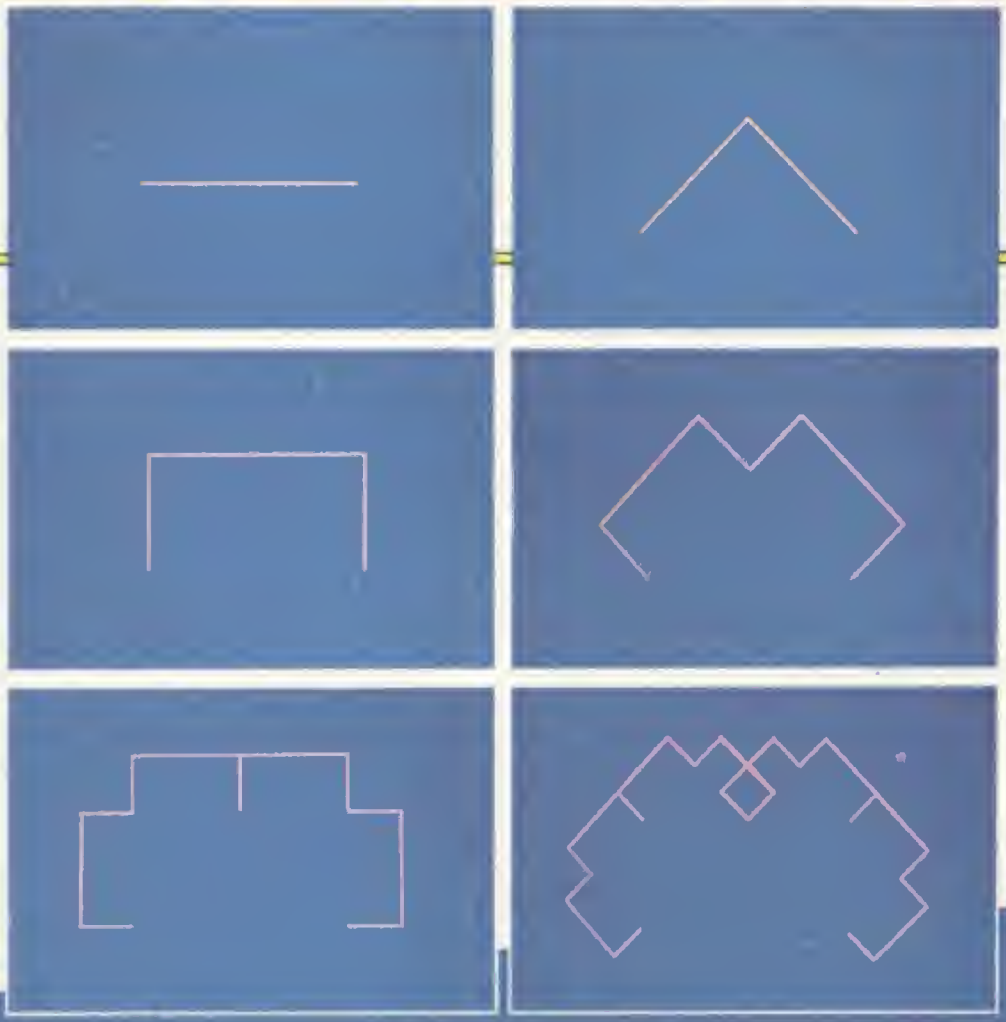
Tomemos como exemplo uma encosta rochosa com 5 km de extensão. Essa medida equivale ao percurso de uma gaivota que voa de uma ponta à outra ou à distância que você realmente andaria seguindo a linha do mar? Há uma diferença considerável entre os dois trajetos, sendo que a rota que acompanha todas as irregularidades é bem maior que a do vôo retilíneo do pássaro.

Mas qual é essa diferença? Suponhamos que temos uma trena bem flexível, capaz de se ajustar a todos os pormenores do terreno. Inicialmente, vamos nos ater aos grandes acidentes, como a foz de um rio ou uma baía. Mas, em busca de maior rigor, passaremos a considerar cada rocha que se projeta ao mar, tornando a costa irregular, e, depois, cada pedra à beira d'água. Por fim, levaremos em conta até os grãos de areia: se você os observar em detalhe, verá que também são irregulares e merecem uma boa inspeção.

Isso parece um exercício absurdo — afinal, ninguém precisa de uma medida desse tipo com tamanha precisão. Mas, por meio do exemplo dado, pudemos constatar algo importante: a exatidão da medida de uma superfície irregular depende do tamanho do instrumento de medida de que dispomos.

A ciência tradicional trabalha com modelos de curvas e superfícies lisas. À medida que as olhamos mais de perto, mais elas adquirem a forma plana — do mesmo modo que a superfície da terra é aproximadamente esférica e nós a percebemos como sendo chata.

Porém, como ficou claro no exemplo da costa rochosa, alguns objetos não se mostram planos nem quando examinados de muito perto. E existem muitos outros exemplos na natureza de configurações que possuem detalhes diferentes em cada dimensão de sua estrutura. Porém, apenas recentemente os cientis-



- FIGURAS GEOMÉTRICAS
- MEDIÇÃO
- AUTO-SEMELHANÇA
- DIMENSÕES FRACIONADAS E GRÁFICOS

tas e matemáticos tomaram consciência da importância do estudo desses objetos e criaram modelos para suas estruturas — uma série de figuras geométricas denominadas *figuras de dimensões fracionadas*, ou *fractais*.

Um dos primeiros pesquisadores a estudá-las foi o norte-americano Benoit Mandelbrot que reconheceu nessas figuras um fecundo campo de trabalho, muito apropriado à concepção de modelos para objetos naturais irregulares.

MEDINDO UM OBJETO

Se você tomar dois pedaços de corda do mesmo tamanho e uni-los por uma das pontas, poderá dizer que obteve

uma cópia do pedaço original com o dobro do tamanho. No caso de uma folha de papel, porém, serão necessárias outras três folhas para se obter uma com o formato da original mas com o dobro da extensão; com uma barra de rapadura, você irá precisar de mais sete cubos.

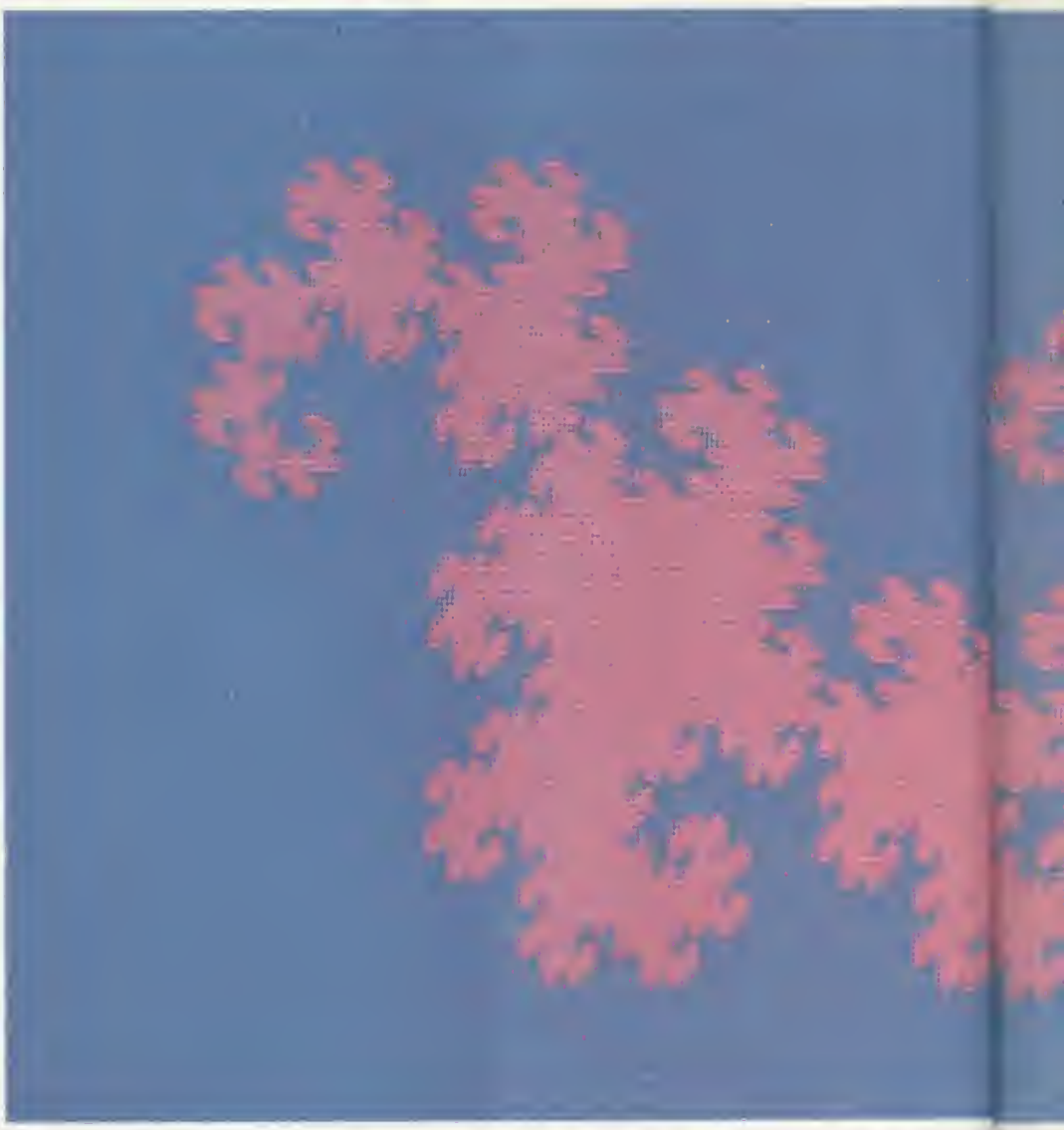
Os números 2, 4 e 8 são parte de uma seqüência formada por meio da multiplicação do 2 por ele mesmo diversas vezes — 2, 2x2 e 2x2x2, ou 2↑1, 2↑2 e 2↑3. A potência à qual está elevado o 2 equivale à dimensão do objeto.

Como vimos, para alcançar o dobro de seu tamanho, um objeto unidimensional (a corda) tem que ser multiplicado por dois; um objeto bidimensional (a folha) deve ser multiplicado por quatro e assim por diante. Suponhamos agora que, para duplicar o tamanho de certo objeto, sejam necessários três exemplares idênticos. Como esse número está entre dois (equivalente a uma dimensão) e quatro (correspondente a duas dimensões), deduzimos que a dimensão do objeto está entre 1 e 2.

À primeira vista, tal situação parece improvável — embora correta, em termos lógicos. Mas o estudo das dimensões fracionadas pode mesmo nos levar a resultados surpreendentes.

O matemático alemão Von Koch foi um dos primeiros estudiosos a desenhar um diagrama representando esse tipo de ocorrência: trata-se da chamada *curva floco de neve* (muito semelhante a um floco de neve visto ao microscópio). Cada um dos lados dessa curva é formado por quatro cópias dela mesma, com um terço do seu tamanho. A dimensão, nesse caso, é um pouco acima de 1,26. Várias formas da natureza, como a própria costa rochosa, podem ser vistas como uma figura dessas.

Como todos os modelos matemáticos, a curva de dimensão fracionada — de definição muito precisa — não é capaz de descrever com perfeição a grande diversidade de objetos naturais, limitando-se a uma aproximação. É, porém, muito útil na explicação de diferentes estruturas — de veias do corpo humano a montanhas, rios ou árvores.



AUTO-SEMELHANÇA

Voltemos um pouco ao exemplo da corda, do papel e da rapadura. Cada um desses objetos poderia ser formado por várias cópias menores idênticas. Estaríamos então diante de um caso de auto-semelhança, onde cada parte do objeto é uma cópia menor do original.

As figuras criadas por Von Koch constituem uma aplicação muito rígida desse princípio, o que explica a extrema regularidade de sua forma. Os exemplares naturais não apresentam necessariamente auto-semelhança — é fácil constatar que um pedaço da casca de uma árvore, por exemplo, não é uma pequena cópia da casca inteira. Entretanto, ele poderia estar em alguma outra parte da casca, pois todos os pedaços se parecem muito. Esse fenômeno, denominado *auto-semelhança estatística*, é muito comum na natureza.

DIMENSÕES FRACIONADAS E GRÁFICOS

Como tantos modelos matemáticos, esse tipo de curva pode ser programado em seu micro para simular a realidade, evidenciando algumas de suas características. No estudo dos fractais, especificamente, a computação gráfica tem sido amplamente utilizada, oferecendo a solução ideal para o problema da geração de figuras irregulares, tais como montanhas, mares e uma série de outros elementos da natureza.

A recursão constitui um excelente meio para a aplicação do princípio da auto-semelhança que envolve a construção de uma figura através de figuras idênticas menores. No caso, uma subrotina principal se encarregará de desenhar os vários elementos idênticos em diferentes tamanhos.

Se você quiser obter uma curva semelhante à curva floco de neve bastará escolher uma figura bem simples e repetila várias vezes, em diversas escalas, adicionando mais e mais detalhes a cada nível de tamanho. Experimente, por exemplo, tomar uma reta e aplicar a regra que prescreve: "toda linha reta deve ser substituída por um par de retas em ângulo reto".

Se você tentar fazer isso em uma folha de papel, verá que a cada passo mais detalhes são introduzidos no desenho.

O programa que apresentamos a seguir faz uma demonstração desse processo. Como todos os gráficos elaborados conforme o modelo de dimensão fracionada, nosso desenho poderia se estender infinitamente. Porém, como não temos uma capacidade gráfica muito grande, os detalhes nos passariam despercebidos, da mesma maneira que as irregularidades de um grão de areia. Portanto, o programa irá parar assim que alcançar um certo nível de recursão.

```
10 BORDER 0: BRIGHT 1: PAPER
0: INK 7: CLS
20 LET mn=2
30 LET c=PI/180
40 LET l=120: LET x=70: LET y
=50: LET an=PI/2
45 PLOT  INVERSE 1; OVER 1;x,
y
50 GOSUB 1000
80 STOP
1000 LET l=l/1.414
1010 IF l<mn THEN  LET l=l*1.41
4: LET x=x+(l*SIN (an)): LET y=
y-()*COS (an)): DRAW x-PEEK 236
77,y-PEEK 23678: RETURN
1020 LET an=an+PI/4: GOSUB 1000
1030 LET an=an-PI/2: GOSUB 1000
1040 LET an=an+PI/4: LET l=l*1.
414: RETURN
```

```
10 PMODE 4,1:PCLS:SCREEN 1,1
20 MN=2
30 C=ATN(1)/45:PI=4*ATN(1)
40 L=120:X=70:Y=140:AN=PI/2
45 LINE-(X,Y),PRESET
50 GOSUB 1000
60 GOTO 60
1000 L=L/1.414
1010 IF L<MN THEN L=L*1.414:X=X
+(L*SIN(AN)):Y=Y+(L*COS(AN)):LI
NE-(X,Y),PSET:RETURN
1020 AN=AN+PI/4:GOSUB 1000
1030 AN=AN-PI/2:GOSUB 1000
1040 AN=AN+PI/4:L=L*414:RETURN
```

```
10  HGR2
20 MN = 2
30 C =  ATN (1) / 45:PI = 4 *
ATN (1)
40 L = 120:X = 70:Y = 140:AN =
PI / 2
45  HPLOT X,Y TO X,Y
50  GOSUB 1000
60  GOTO 60
1000 L = L / 1.414
1010  IF L < MN THEN L = L * 1.
414:X = X + (L *  SIN (AN)):Y =
 Y + (L *  COS (AN)): HCOLOR= 3
: HPLOT  TO X,Y: RETURN
```

```
1020 AN = AN + PI / 4: GOSUB 10
00
1030 AN = AN - PI / 2: GOSUB 10
00
1040 AN = AN + PI / 4:L = L * 1
.414: RETURN
```

```
10 SCREEN 2
20 MN=2
30 C=ATN(1)/45:PI=4*ATN(1)
40 L=120:X=70:Y=140:AN=PI/2
45 LINE-(X,Y),4
50 GOSUB 1000
60 GOTO 60
1000 L=L/1.414
1010 IF L<MN THEN L=L*1.414:X=X
+(L*SIN(AN)):Y=Y+(L*COS(AN)):LI
NE -(X,Y),11:RETURN
1020 AN=AN+PI/4:GOSUB 1000
1030 AN=AN-PI/2:GOSUB 1000
1040 AN=AN+PI/4:L=L*1.414:RETUR
N
```

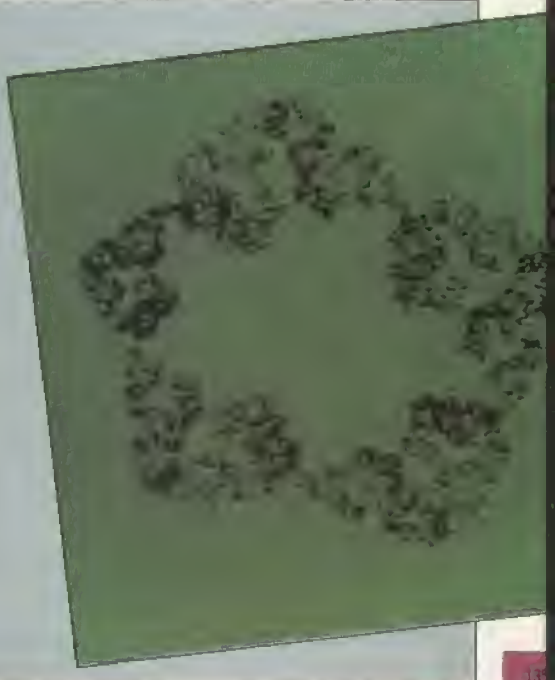
O programa desenhará na tela uma curva em forma de C. Inicialmente, ele traça uma reta, que logo é substituída por um par de retas em ângulo reto. A sub-rotina entre as linhas 1000 e 1040, que usa **SIN** e **COS** para construir os ângulos necessários, é chamada na linha 50. Em um processo recursivo, ela chama a si mesma repetidamente, para substituir as linhas retas por pares de retas perpendiculares.

A variável da linha 20 contém o comprimento da linha mais curta — a recursão será encerrada quando alcançar esse valor. Tente mudá-lo e observe o efeito resultante: valores menores aumen-

MODELOS EXPERIMENTAIS

A principal diferença entre as figuras de dimensão fracionada geradas no computador e as formas da natureza está na irregularidade e desordem destas últimas.

No próximo artigo sobre fractais, veremos como acrescentar elementos aleatórios ao processo de construção das figuras, aproximando-as mais dos modelos naturais. Um dos programas, um gerador de figuras, permite que se introduza a semente geradora do desenho — ou seja, uma infinidade de formas poderá ser criada simplesmente variando-se a informação inicial. Assim, a cada nova execução do programa, você obterá uma curva diferente. Uma das muitas possibilidades é mostrada na ilustração à direita.



tam o número de níveis de recursão e, em conseqüência, o tempo de execução. Mas, se introduzirmos valores maiores, o programa será acelerado e poderemos visualizar melhor o processo de elaboração do gráfico.

Se você acompanhar o desenvolvimento da figura na tela com atenção, verificará que cada parte da curva se parece com a própria curva (princípio da auto-semelhança). Apesar disso, não é fácil antever, logo no início, a forma que ela irá adquirir.

CURVA DO DRAGÃO

Quando trabalhamos com curvas de dimensão fracionada, poucas linhas de programa são necessárias para produzir uma bela figura — sobretudo se utilizarmos a recursão. Digite e execute o próximo programa para obter uma curva em forma de dragão.

No programa anterior, cada reta era substituída por um par de retas perpendiculares, traçadas sempre em uma mesma direção. Neste programa, porém, pares de retas são colocados alternadamente de um e de outro lado de cada reta, até formar a figura final.

```
10 BORDER 0: PAPER 0: INK 7:
CLS : CLEAR 30000: LET S=
32767: DEF FN A(X)=(X/8-INT (
X/8))*8
20 LET MN=1: LET A=0
30 LET C=ATN (1)/45: DIM S(10
): DIM C(10)
40 FOR I=1 TO 8: LET S(I)=SIN
A
50 LET C(I)=COS A: LET A=A+PI
/4: NEXT I
60 LET L=128: LET X=52: LET Y
=80: LET T=-1: POKE S,T+1:
LET S=S-1
65 DRAW  INVERSE 1; OVER 1;X-
PEEK 23677,Y-PEEK 23678
70 GOSUB 1000
80 STOP
1000 LET L=L/1.414
1010 IF L<MN THEN  LET L=L*1.41
4: LET X=X+(L*C(I)): LET Y=Y-(L
*S(I)): DRAW X-PEEK 23677,Y-PEE
K 23678: RETURN
1020 LET I=FN A(I+T): POKE S,T+
1: LET S=S-1: LET T=1: GOSUB 10
00: LET S=S+1: LET T=(PEEK S)-1
1030 LET I=FN A(I-2*T): POKE S,
T+1: LET S=S-1: LET T=-1: GOSUB
 1000
1040 LET S=S+1: LET T=(PEEK S)-
1: LET I=FN A(I+T): LET L=L*1.4
14: RETURN
```

```
10 PMODE 4:PCLS:SCREEN 1,1:CLEA
R 200,30000:S=32767
20 MN=1
30 C=ATN(1)/45:PI=4*ATN(1)
40 FOR I=0 TO 7:S(I)=SIN(A)
50 C(I)=COS(A):A=A+PI/4:NEXT
60 L=128:X=52:Y=80:T=-1:POKE S,
T+1:S=S-1
65 LINE -(X,Y),PRESET
70 GOSUB 1000
80 GOTO 80
1000 L=L/1.414
1010 IF L<MN THEN L=L*1.414:X=X
+(L*C(I)):Y=Y-(L*(S(I)):LINE-(X
,Y),PSET:RETURN
1020 I=(I+T) AND 7:POKE S,T+1:S
=S-1:T=1:GOSUB 1000:S=S+1:T=PEE
K(S)-1
1030 I=(I-2*T)AND7:POKE S,T+1:S
=S-1:T=-1:GOSUB 1000
1040 S=S+1:T=PEEK(S)-1:I=(I+T)
AND 7:L=L*1.414:RETURN
```

```
10 S = 900
20  HGR2 :MN = 1
30 C =  ATN (1) / 45:PI = 4 *
ATN (1)
40  FOR I = 0 TO 7:S(I) =  SIN
(A)
50 C(I) =  COS (A):A = A + PI /
 4: NEXT
60 L = 128:X = 52:Y = 80:T =  -
 1: POKE S,T + 1:S = S - 1
65  HCOLOR= 3: HPLOT X,Y TO X,Y

70  GOSUB 1000
80  GOTO 80
1000 L = L / 1.414
1010  IF L < MN THEN L = L * 1.
414:X = X + (L * C(I)):Y = Y -
(L * S(I)): HPLOT  TO X,Y: RETU
RN
1020 I = ((I + T) - 8 *  INT ((
I + T) / 8)): POKE S,T + 1:S =
S - 1:T = 1: GOSUB 1000:S = S +
 1:T =  PEEK (S) - 1
1030 I = ((I - 2 * T) - 8 *  IN
T ((I - 2 * T) / 8)): POKE S,T
+ 1:S = S - 1:T =  - 1: GOSUB 1
000
1040 S = S + 1:T =  PEEK (S) -
1:I = ((I + T) AND 7):L = L * 1
.414: RETURN
```

```
10 SCREEN 2:CLEAR 200,50000!:S=
52767!
20 MN=1
30 C=ATN(1)/45:PI=4*ATN(1)
40 FOR I=0 TO 7:S(I)=SIN(A)
50 C(I)=COS(A):A=A+PI/4:NEXT
60 L=128:X=52:Y=80:T=-1:POKE S,
T+1:S=S-1
65 LINE -(X,Y),4
70 GOSUB 1000
80 GOTO 80
1000 L=L/1.414
1010 IF L<MN THEN L=L*1.414:X=X
+(L*C(I)):Y=Y-(L*S(I)):LINE -(X
,Y),11:RETURN
1020 I=(I+T) AND 7:POKE S,T+1:S
=S-1:T=1:GOSUB 1000:S=S+1:T=PEE
K(S)-1
1030 I=(I-2*T) AND 7:POKE S,T+1
:S=S-1:T=-1:GOSUB 1000
1040 S=S+1:T=PEEK(S)-1:I=(I+T)A
ND 7:L=L*1.414:RETURN
```

Existe alguma aplicação prática para os fractais?

Os fractais são uma descoberta relativamente recente da matemática e das ciências da computação. Assim, há muito a ser investigado acerca de seu funcionamento e as aplicações práticas ainda são bastante restritas. Os fractais oferecem aos artistas e cientistas a possibilidade de representar as formas irregulares, típicas da natureza, e já têm marcado sua presença no mundo das artes gráficas, cinema, televisão etc. Grandes empresas de televisão, por exemplo, estão utilizando fractais randômicos para gerar em computador fantásticas paisagens e cenários, terrestres ou extraterrestres. A simulação fractal apresenta a vantagem de, com pequenas mudanças nos parâmetros do programa, produzir efeitos totalmente diferentes na tela.

Outra aplicação prática interessante dos fractais é o desenho de cenários complexos em simuladores de vôo. A Força Aérea norte-americana já dispõe de programas que permitem gerar o cenário de uma batalha aérea entre montanhas escarpadas, sobre desertos etc., para treinar o piloto de helicópteros ou caças.

Esse programa difere do anterior quanto à regra de formação do desenho e, também, quanto à velocidade de execução. Em vez de elaborar o ângulo cada vez que um elemento é traçado, o programa calcula o seno e o co-seno dos ângulos (linhas 40 e 50) e coloca os valores obtidos em duas matrizes. Recuperar esses valores em uma matriz é bem mais rápido que calculá-los sempre que uma linha é reconstruída.

A curva em C e a curva do dragão não são apenas curiosidades matemáticas. Além de muito bonitas, elas constituem um interessante objeto de estudo. Procure, por exemplo, alterar alguns valores dentro dos dois programas. Você obterá, então, um estoque inesgotável de novas formas.

LINHA	FABRICANTE	MODELO
Apple II +	Appletronica	Thor 2010
Apple II +	CCE	MC-4000 Exato
Apple II +	CPA	Absolutus
Apple II +	CPA	Polaris
Apple II +	Digitus	DGT-AP
Apple II +	Dismac	D-8100
Apple II +	ENIAC	ENIAC II
Apple II +	Franklin	Franklin
Apple II +	Houston	Houston AP
Apple II +	Magnex	DM II
Apple II +	Maxitronica	MX-2001
Apple II +	Maxitronica	MX-48
Apple II +	Maxitronica	MX-64
Apple II +	Maxitronica	Maxitronic I
Apple II +	Microcraft	Craft II Plus
Apple II +	Milmar	Apple II Plus
Apple II +	Milmar	Apple Master
Apple II +	Milmar	Apple Senior
Apple II +	Omega	MC-400
Apple II +	Polymax	Maxxi
Apple II +	Polymax	Poly Plus
Apple II +	Spectrum	Microengenho I
Apple II +	Spectrum	Spectrum ed
Apple II +	Suporte	Venus II
Apple II +	Sycomig	SIC I
Apple II +	Unitron	AP II
Apple II +	Victor do Brasil	Elppa II Plus
Apple II +	Victor do Brasil	Elppa Jr.
Apple IIe	Microcraft	Craft IIe
Apple IIe	Microdigital	TK-3000 IIe
Apple IIe	Spectrum	Microengenho II
MSX	Gradiente	Expert GPC-1
MSX	Sharp	Hotbit HB-8000
Sinclair Spectrum	Microdigital	TK-90X
Sinclair Spectrum	Timex	Timex 2000
Sinclair ZX-81	Apply	Apply 300
Sinclair ZX-81	Engebras	AS-1000
Sinclair ZX-81	Filcres	NEZ-8000
Sinclair ZX-81	Microdigital	TK-82C
Sinclair ZX-81	Microdigital	TK-83
Sinclair ZX-81	Microdigital	TK-85
Sinclair ZX-81	Prologica	CP-200
Sinclair ZX-81	Ritas	Ringo R-470
Sinclair ZX-81	Timex	Timex 1000
Sinclair ZX-81	Timex	Timex 1500
TRS-80 Mod. I	Dismac	D-8000
TRS-80 Mod. I	Dismac	D-8001/2
TRS-80 Mod. I	LNW	LNW-80
TRS-80 Mod. I	Video Genie	Video Genie I
TRS-80 Mod.III	Digitus	DGT-100
TRS-80 Mod.III	Digitus	DGT-1000
TRS-80 Mod.III	Kemitron	Naja 800
TRS-80 Mod.III	Prologica	CP-300
TRS-80 Mod.III	Prologica	CP-500
TRS-80 Mod.III	Sysdata	Sysdata III
TRS-80 Mod.III	Sysdata	Sysdata Jr.
TRS-80 Mod.IV	Multix	MX-Compacto
TRS-80 Mod.IV	Sysdata	Sysdata IV
TRS-Color	Codimex	CS-6508
TRS-Color	Dynacom	MX-1600
TRS-Color	LZ	Color 64
TRS-Color	Microdigital	TKS-800
TRS-Color	Prologica	CP-400

FABRICANTE	MODELO	PAÍS	LINHA
Appletronica	Thor 2010	Brasil	Apple II +
Apply	Apply 300	Brasil	Sinclair ZX-81
CCE	MC-4000 Exato	Brasil	Apple II +
CPA	Absolutus	Brasil	Apple II +
CPA	Polaris	Brasil	Apple II +
Codimex	CS-6508	Brasil	TRS-Color
Digitus	DGT-100	Brasil	TRS-80 Mod.III
Digitus	DGT-1000	Brasil	TRS-80 Mod.III
Digitus	DGT-AP	Brasil	Apple II +
Dismac	D-8000	Brasil	TRS-80 Mod. I
Dismac	D-8001/2	Brasil	TRS-80 Mod. I
Dismac	D-8100	Brasil	Apple II +
Dynacom	MX-1600	Brasil	TRS-Color
ENIAC	ENIAC II	Brasil	Apple II +
Engebras	AS-1000	Brasil	Sinclair ZX-81
Filcres	NEZ-8000	Brasil	Sinclair ZX-81
Franklin	Franklin	USA	Apple II +
Gradiente	Expert GPC1	Brasil	MSX
Houston	Houston AP	Brasil	Apple II +
Kemitron	Naja 800	Brasil	TRS-80 Mod.III
LNW	LNW-80	USA	TRS-80 Mod. I
LZ	Color 64	Brasil	TRS-Color
Magnex	DM II	Brasil	Apple II +
Maxitronica	MX-2001	Brasil	Apple II +
Maxitronica	MX-48	Brasil	Apple II +
Maxitronica	MX-64	Brasil	Apple II +
Maxitronica	Maxitronic I	Brasil	Apple II +
Microcraft	Craft II Plus	Brasil	Apple II +
Microcraft	Craft IIe	Brasil	Apple IIe
Microdigital	TK-3000 IIe	Brasil	Apple IIe
Microdigital	TK-82C	Brasil	Sinclair ZX-81
Microdigital	TK-83	Brasil	Sinclair ZX-81
Microdigital	TK-85	Brasil	Sinclair ZX-81
Microdigital	TK-90X	Brasil	Sinclair Spectrum
Microdigital	TKS-800	Brasil	TRS-Color
Milmar	Apple II Plus	Brasil	Apple II +
Milmar	Apple Master	Brasil	Apple II +
Milmar	Apple Senior	Brasil	Apple II +
Multix	MX-Compacto	Brasil	TRS-80 Mod.IV
Omega	MC-400	Brasil	Apple II +
Polymax	Maxxi	Brasil	Apple II +
Polymax	Poly Plus	Brasil	Apple II +
Prologica	CP-200	Brasil	Sinclair ZX-81
Prologica	CP-300	Brasil	TRS-80 Mod.III
Prologica	CP-400	Brasil	TRS-Color
Prologica	CP-500	Brasil	TRS-80 Mod.III
Ritas	Ringo R-470	Brasil	Sinclair ZX-81
Sharp	Hotbit HB-8000	Brasil	MSX
Spectrum	Microengenho I	Brasil	Apple II +
Spectrum	Microengenho II	Brasil	Apple IIe
Spectrum	Spectrum ed	Brasil	Apple II +
Suporte	Venus II	Brasil	Apple II +
Sycomig	SIC I	Brasil	Apple II +
Sysdata	Sysdata III	Brasil	TRS-80 Mod.III
Sysdata	Sysdata IV	Brasil	TRS-80 Mod.IV
Sysdata	Sysdata Jr.	Brasil	TRS-80 Mod.III
Timex	Timex 1000	USA	Sinclair ZX-81
Timex	Timex 1500	USA	Sinclair ZX-81
Timex	Timex 2000	USA	Sinclair Spectrum
Unitron	AP II	Brasil	Apple II +
Victor do Brasil	Elppa II Plus	Brasil	Apple II +
Victor do Brasil	Elppa Jr.	Brasil	Apple II +
Video Genie	Video Genie I	USA	TRS-80 Mod. I

INPUT foi especialmente projetado para microcomputadores compatíveis com as sete principais linhas existentes no mercado.
Os blocos de textos e listagens de programas aplicados apenas a determinadas linhas de micros podem ser identificados por meio dos seguintes símbolos:


Sinclair ZX-81


TRS-80


TK-2000


MSX


Spectrum


TRS-Color


Apple II

Quando o emblema for seguido de uma faixa, então tanto o texto como os programas que se seguem passam a ser específicos para a linha indicada.

NO PRÓXIMO NÚMERO

PROGRAMAÇÃO BASIC
A tela de textos do MSX. Desenho das letras. A função BASE. Tabela de Padrões. Como criar um novo conjunto de caracteres.

APLICAÇÕES
O uso do microcomputador no planejamento de ambientes. Como tirar as medidas. Desenhos e escala. O giro das peças.

PROGRAMAÇÃO BASIC
Simulação de imagens com a técnica dos fractais. Simetria e assimetria. Programa gerador de figuras. Teste do programa.

PROGRAMAÇÃO BASIC
Bits e bytes em BASIC. Mudança de um bit. Combinações. **AND, OR** e **NOT.**

CURSO PRÁTICO 69 DE PROGRAMAÇÃO DE COMPUTADORES

INFO BASIC

PROGRAMAÇÃO BASIC - PROGRAMAÇÃO DE JOGOS - CÓDIGO DE MÁQUINA